

# 12th Innovations in Theoretical Computer Science Conference

ITCS 2021, January 6–8, 2021, Virtual Conference

Edited by

James R. Lee



*Editors*

**James R. Lee**

University of Washington, Seattle, USA  
jrl@cs.washington.edu

*ACM Classification 2012*

Mathematics of computing; Theory of computation

**ISBN 978-3-95977-177-1**

*Published online and open access by*

Schloss Dagstuhl – Leibniz-Zentrum für Informatik GmbH, Dagstuhl Publishing, Saarbrücken/Wadern, Germany. Online available at <https://www.dagstuhl.de/dagpub/978-3-95977-177-1>.

*Publication date*

February, 2021

*Bibliographic information published by the Deutsche Nationalbibliothek*

The Deutsche Nationalbibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data are available in the Internet at <https://portal.dnb.de>.

*License*

This work is licensed under a Creative Commons Attribution 4.0 International license (CC-BY 4.0):  
<https://creativecommons.org/licenses/by/4.0/legalcode>.



In brief, this license authorizes each and everybody to share (to copy, distribute and transmit) the work under the following conditions, without impairing or restricting the authors' moral rights:

- Attribution: The work must be attributed to its authors.

The copyright is retained by the corresponding authors.

Digital Object Identifier: 10.4230/LIPIcs.ITCS.2021.0

**ISBN 978-3-95977-177-1**

**ISSN 1868-8969**

**<https://www.dagstuhl.de/lipics>**



## LIPICs – Leibniz International Proceedings in Informatics

LIPICs is a series of high-quality conference proceedings across all fields in informatics. LIPICs volumes are published according to the principle of Open Access, i.e., they are available online and free of charge.

### *Editorial Board*

- Luca Aceto (*Chair*, Gran Sasso Science Institute and Reykjavik University)
- Christel Baier (TU Dresden)
- Mikolaj Bojanczyk (University of Warsaw)
- Roberto Di Cosmo (INRIA and University Paris Diderot)
- Javier Esparza (TU München)
- Meena Mahajan (Institute of Mathematical Sciences)
- Dieter van Melkebeek (University of Wisconsin-Madison)
- Anca Muscholl (University Bordeaux)
- Luke Ong (University of Oxford)
- Catuscia Palamidessi (INRIA)
- Thomas Schwentick (TU Dortmund)
- Raimund Seidel (Saarland University and Schloss Dagstuhl – Leibniz-Zentrum für Informatik)

**ISSN 1868-8969**

**<https://www.dagstuhl.de/lipics>**



## ■ Contents

Preface	
<i>James R. Lee</i> .....	0:xi
ITCS 2021 Conference Organization	
.....	0:xiii–0:xiv

## Papers

The Entropy of Lies: Playing Twenty Questions with a Liar	
<i>Yuval Dagan, Yuval Filmus, Daniel Kane, and Shay Moran</i> .....	1:1–1:16
Comparing Computational Entropies Below Majority (Or: When Is the Dense Model Theorem False?)	
<i>Russell Impagliazzo and Sam McGuire</i> .....	2:1–2:20
Algorithmic Persuasion with Evidence	
<i>Martin Hoefer, Pasin Manurangsi, and Alexandros Psomas</i> .....	3:1–3:20
The Complexity of Finding Fair Independent Sets in Cycles	
<i>Ishay Haviv</i> .....	4:1–4:14
Sharp Threshold Rates for Random Codes	
<i>Venkatesan Guruswami, Jonathan Mosheiff, Nicolas Resch, Shashwat Silas, and Mary Wootters</i> .....	5:1–5:20
Simple Heuristics Yield Provable Algorithms for Masked Low-Rank Approximation	
<i>Cameron Musco, Christopher Musco, and David P. Woodruff</i> .....	6:1–6:20
Pseudorandom Generators for Unbounded-Width Permutation Branching Programs	
<i>William M. Hoza, Edward Pyne, and Salil Vadhan</i> .....	7:1–7:20
Pipeline Interventions	
<i>Eshwar Ram Arunachaleswaran, Sampath Kannan, Aaron Roth, and Juba Ziani</i> ..	8:1–8:20
A Polynomial Degree Bound on Equations for Non-Rigid Matrices and Small Linear Circuits	
<i>Mrinal Kumar and Ben Lee Volk</i> .....	9:1–9:9
The Strongish Planted Clique Hypothesis and Its Consequences	
<i>Pasin Manurangsi, Aviad Rubinfeld, and Tselil Schramm</i> .....	10:1–10:21
Sample Efficient Identity Testing and Independence Testing of Quantum States	
<i>Nengkun Yu</i> .....	11:1–11:20
Understanding the Relative Strength of QBF CDCL Solvers and QBF Resolution	
<i>Olaf Beyersdorff and Benjamin Böhm</i> .....	12:1–12:20
The Quantum Supremacy Tsirelson Inequality	
<i>William Kretschmer</i> .....	13:1–13:13

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee



Leibniz International Proceedings in Informatics  
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Approximately Strategyproof Tournament Rules in the Probabilistic Setting <i>Kimberly Ding and S. Matthew Weinberg</i> .....	14:1–14:20
Even the Easiest(?) Graph Coloring Problem Is Not Easy in Streaming! <i>Anup Bhattacharya, Arijit Bishnu, Gopinath Mishra, and Anannya Upasana</i> .....	15:1–15:19
The Variable-Processor Cup Game <i>William Kuszmaul and Alek Westover</i> .....	16:1–16:20
Comparison Graphs: A Unified Method for Uniformity Testing <i>Uri Meir</i> .....	17:1–17:20
Circular Trace Reconstruction <i>Shyam Narayanan and Michael Ren</i> .....	18:1–18:18
Self-Testing of a Single Quantum Device Under Computational Assumptions <i>Tony Metger and Thomas Vidick</i> .....	19:1–19:12
Polynomial-Time Trace Reconstruction in the Low Deletion Rate Regime <i>Xi Chen, Anindya De, Chin Ho Lee, Rocco A. Servedio, and Sandip Sinha</i> .....	20:1–20:20
Metrical Service Systems with Transformations <i>Sébastien Bubeck, Niv Buchbinder, Christian Coester, and Mark Sellke</i> .....	21:1–21:20
Tight Hardness Results for Training Depth-2 ReLU Networks <i>Surbhi Goel, Adam Klivans, Pasin Manurangsi, and Daniel Reichman</i> .....	22:1–22:14
A Largish Sum-Of-Squares Implies Circuit Hardness and Derandomization <i>Pranjal Dutta, Nitin Saxena, and Thomas Thierauf</i> .....	23:1–23:21
Circuit Depth Reductions <i>Alexander Golovnev, Alexander S. Kulikov, and R. Ryan Williams</i> .....	24:1–24:20
Dynamic Inference in Probabilistic Graphical Models <i>Weiming Feng, Kun He, Xiaoming Sun, and Yitong Yin</i> .....	25:1–25:20
Theorems of KKL, Friedgut, and Talagrand via Random Restrictions and Log-Sobolev Inequality <i>Esty Kelman, Subhash Khot, Guy Kindler, Dor Minzer, and Muli Safra</i> .....	26:1–26:17
On Rich 2-to-1 Games <i>Mark Braverman, Subhash Khot, and Dor Minzer</i> .....	27:1–27:20
Distributed Quantum Proofs for Replicated Data <i>Pierre Fraigniaud, François Le Gall, Harumichi Nishimura, and Ami Paz</i> .....	28:1–28:20
On Basing Auxiliary-Input Cryptography on NP-Hardness via Nonadaptive Black-Box Reductions <i>Mikito Nanashima</i> .....	29:1–29:15
Spoofing Linear Cross-Entropy Benchmarking in Shallow Quantum Circuits <i>Boaz Barak, Chi-Ning Chou, and Xun Gao</i> .....	30:1–30:20
On the Complexity of Isomorphism Problems for Tensors, Groups, and Polynomials I: Tensor Isomorphism-Completeness <i>Joshua A. Grochow and Youming Qiao</i> .....	31:1–31:19
Bounds on the QAC <sup>0</sup> Complexity of Approximating Parity <i>Gregory Rosenthal</i> .....	32:1–32:20

Query Complexity Lower Bounds for Local List-Decoding and Hard-Core Predicates (Even for Small Rate and Huge Lists)	
<i>Noga Ron-Zewi, Ronen Shaltiel, and Nithin Varma</i>	33:1–33:18
Is the Space Complexity of Planted Clique Recovery the Same as That of Detection?	
<i>Jay Mardia</i>	34:1–34:17
Algorithms and Hardness for Multidimensional Range Updates and Queries	
<i>Joshua Lau and Angus Ritossa</i>	35:1–35:20
Two Combinatorial MA-Complete Problems	
<i>Dorit Aharonov and Alex B. Grilo</i>	36:1–36:20
Delegated Stochastic Probing	
<i>Curtis Bechtel and Shaddin Dughmi</i>	37:1–37:19
Explicit SoS Lower Bounds from High-Dimensional Expanders	
<i>Irit Dinur, Yuval Filmus, Prahladh Harsha, and Madhur Tulsiani</i>	38:1–38:16
Lower Bounds on the Running Time of Two-Way Quantum Finite Automata and Sublogarithmic-Space Quantum Turing Machines	
<i>Zachary Remscrem</i>	39:1–39:20
On the Complexity of $\#CSP^d$	
<i>Jiabao Lin</i>	40:1–40:10
Interactive Proofs for Verifying Machine Learning	
<i>Shafi Goldwasser, Guy N. Rothblum, Jonathan Shafer, and Amir Yehudayoff</i>	41:1–41:19
Ordered Graph Limits and Their Applications	
<i>Omri Ben-Eliezer, Eldar Fischer, Amit Levi, and Yuichi Yoshida</i>	42:1–42:20
Error Correcting Codes for Uncompressed Messages	
<i>Ofer Grossman and Justin Holmgren</i>	43:1–43:18
Total Functions in the Polynomial Hierarchy	
<i>Robert Kleinberg, Oliver Korten, Daniel Mitropolsky, and Christos Papadimitriou</i>	44:1–44:18
Relaxing Common Belief for Social Networks	
<i>Noah Burrell and Grant Schoenebeck</i>	45:1–45:20
Tiered Random Matching Markets: Rank Is Proportional to Popularity	
<i>Itai Ashlagi, Mark Braverman, Amin Saberi, Clayton Thomas, and Geng Zhao</i>	46:1–46:16
Black-Box Uselessness: Composing Separations in Cryptography	
<i>Geoffroy Couteau, Pooya Farshim, and Mohammad Mahmoody</i>	47:1–47:20
Pseudobinominality of the Sticky Random Walk	
<i>Venkatesan Guruswami and Vinayak M. Kumar</i>	48:1–48:19
Robust Quantum Entanglement at (Nearly) Room Temperature	
<i>Lior Eldar</i>	49:1–49:20
Time-Space Lower Bounds for Simulating Proof Systems with Quantum and Randomized Verifiers	
<i>Abhijit S. Mudigonda and R. Ryan Williams</i>	50:1–50:20

Online Search with a Hint <i>Spyros Angelopoulos</i> .....	51:1–51:16
Sequential Defaulting in Financial Networks <i>Pál András Papp and Roger Wattenhofer</i> .....	52:1–52:20
No Quantum Speedup over Gradient Descent for Non-Smooth Convex Optimization <i>Ankit Garg, Robin Kothari, Praneeth Netrapalli, and Suhail Sherif</i> .....	53:1–53:20
Quantum Versus Randomized Communication Complexity, with Efficient Players <i>Uma Girish, Ran Raz, and Avishay Tal</i> .....	54:1–54:20
Agnostic Learning with Unknown Utilities <i>Kush Bhatia, Peter L. Bartlett, Anca D. Dragan, and Jacob Steinhardt</i> .....	55:1–55:20
On Distributed Differential Privacy and Counting Distinct Elements <i>Lijie Chen, Badih Ghazi, Ravi Kumar, and Pasin Manurangsi</i> .....	56:1–56:18
A Generalized Matching Reconfiguration Problem <i>Noam Solomon and Shay Solomon</i> .....	57:1–57:20
Sensitivity Analysis of the Maximum Matching Problem <i>Yuichi Yoshida and Samson Zhou</i> .....	58:1–58:20
Computational Complexity of the Hylland-Zeckhauser Scheme for One-Sided Matching Markets <i>Vijay V. Vazirani and Mihalis Yannakakis</i> .....	59:1–59:19
An $O(N)$ Time Algorithm for Finding Hamilton Cycles with High Probability <i>Rajko Nenadov, Angelika Steger, and Pascal Su</i> .....	60:1–60:17
Communication Memento: Memoryless Communication Complexity <i>Srinivasan Arunachalam and Supartha Podder</i> .....	61:1–61:20
Relative Lipschitzness in Extragradient Methods and a Direct Recipe for Acceleration <i>Michael B. Cohen, Aaron Sidford, and Kevin Tian</i> .....	62:1–62:18
Training (Overparametrized) Neural Networks in Near-Linear Time <i>Jan van den Brand, Binghui Peng, Zhao Song, and Omri Weinstein</i> .....	63:1–63:15
A New Connection Between Node and Edge Depth Robust Graphs <i>Jeremiah Blocki and Mike Cinkoske</i> .....	64:1–64:18
Towards Local Testability for Quantum Coding <i>Anthony Leverrier, Vivien Londe, and Gilles Zémor</i> .....	65:1–65:11
Complete Problems for Multi-Pseudodeterministic Computations <i>Peter Dixon, A. Pavan, and N. V. Vinodchandran</i> .....	66:1–66:16
Online Paging with a Vanishing Regret <i>Yuval Emek, Shay Kutten, and Yangguang Shi</i> .....	67:1–67:20
Differentially Oblivious Turing Machines <i>Ilan Komargodski and Elaine Shi</i> .....	68:1–68:19
Quantitative Correlation Inequalities via Semigroup Interpolation <i>Anindya De, Shivam Nadimpalli, and Rocco A. Servedio</i> .....	69:1–69:20

Shrinkage of Decision Lists and DNF Formulas <i>Benjamin Rossman</i> .....	70:1–70:14
Block Rigidity: Strong Multiplayer Parallel Repetition Implies Super-Linear Lower Bounds for Turing Machines <i>Kunal Mittal and Ran Raz</i> .....	71:1–71:15
Lower Bounds for Off-Chain Protocols: Exploring the Limits of Plasma <i>Stefan Dziembowski, Grzegorz Fabiański, Sebastian Faust, and Siavash Riahi</i> .....	72:1–72:20
Majorizing Measures for the Optimizer <i>Sander Borst, Daniel Dadush, Neil Olver, and Makrand Sinha</i> .....	73:1–73:20
Randomness and Fairness in Two-Sided Matching with Limited Interviews <i>Hedyeh Beyhaghi and Éva Tardos</i> .....	74:1–74:18
Counterexamples to the Low-Degree Conjecture <i>Justin Holmgren and Alexander S. Wein</i> .....	75:1–75:9
High-Entropy Dual Functions and Locally Decodable Codes (Extended Abstract) <i>Jop Briët and Farrokh Labib</i> .....	76:1–76:2
Buying Data over Time: Approximately Optimal Strategies for Dynamic Data-Driven Decisions <i>Nicole Immorlica, Ian A. Kash, and Brendan Lucier</i> .....	77:1–77:14
Learning and Strongly Truthful Multi-Task Peer Prediction: A Variational Approach <i>Grant Schoenebeck and Fang-Yi Yu</i> .....	78:1–78:20
Distributed Load Balancing: A New Framework and Improved Guarantees <i>Sara Ahmadian, Allen Liu, Binghui Peng, and Morteza Zadimoghaddam</i> .....	79:1–79:20
Erasure-Resilient Sublinear-Time Graph Algorithms <i>Amit Levi, Ramesh Krishnan S. Pallavoor, Sofya Raskhodnikova, and Nithin Varma</i> .....	80:1–80:20
How to Sell Information Optimally: An Algorithmic Study <i>Yang Cai and Grigoris Velezgas</i> .....	81:1–81:20
Computation over the Noisy Broadcast Channel with Malicious Parties <i>Klim Efremenko, Gillat Kol, Dmitry Paramonov, and Raghuvansh R. Saxena</i> ....	82:1–82:19
Sampling Arborescences in Parallel <i>Nima Anari, Nathan Hu, Amin Saberi, and Aaron Schild</i> .....	83:1–83:18
Non-Quasi-Linear Agents in Quasi-Linear Mechanisms (Extended Abstract) <i>Moshe Babaioff, Richard Cole, Jason Hartline, Nicole Immorlica, and Brendan Lucier</i> .....	84:1–84:1
A Model for Ant Trail Formation and its Convergence Properties (Extended Abstract) <i>Moses Charikar, Shivam Garg, Deborah M. Gordon, and Kirankumar Shiragur</i> ...	85:1–85:2
Unknown I.I.D. Prophets: Better Bounds, Streaming Algorithms, and a New Impossibility (Extended Abstract) <i>José Correa, Paul Dütting, Felix Fischer, Kevin Schewior, and Bruno Ziliotto</i> ....	86:1–86:1

Complexity Measures on the Symmetric Group and Beyond (Extended Abstract) <i>Neta Dafni, Yuval Filmus, Noam Lifshitz, Nathan Lindzey, and Marc Vinyals</i> . . . .	87:1–87:5
Batching and Optimal Multi-Stage Bipartite Allocations (Extended Abstract) <i>Yiding Feng and Rad Niazadeh</i> . . . . .	88:1–88:1
Shrinkage Under Random Projections, and Cubic Formula Lower Bounds for AC0 (Extended Abstract) <i>Yuval Filmus, Or Meir, and Avishay Tal</i> . . . . .	89:1–89:7



## ■ Preface

The papers in this volume were presented at the 12th Innovations in Theoretical Computer Science (ITCS 2021) conference. The conference was held *online* from January 6–8, 2021. ITCS seeks to promote research that carries a strong conceptual message, for instance, introducing a new concept or model, opening a new line of inquiry within traditional or cross-interdisciplinary areas, introducing new techniques, or making novel connections between existing areas and ideas. The conference format is single-session and aims to promote the exchange of ideas between different areas of theoretical computer science and with other disciplines. A record 214 submissions were received and, of these, the program committee selected 89 papers. The submission pool was strong across the board, and it's very fulfilling to see the tradition of ITCS continue to grow.

The program committee consisted of 37 members (in addition to the chair): Andris Ambainis, University of Latvia; Nima Anari, Stanford; Elette Boyle, IDC Herzliya; Mark Braverman, Princeton; Sebastien Bubeck, Microsoft Research; Claire Mathieu, CNRS, Paris; Edith Cohen, Google; Anindya De, University of Pennsylvania; Uriel Feige, Weizmann Institute; Kira Goldner, Columbia; Monika Henzinger, University of Vienna; Maurice Herlihy, Brown; Sam Hopkins, UC Berkeley and MIT; Tali Kaufman, Bar-Ilan University; Adam Klivans, UT Austin; Gillat Kol, Princeton; Alexandra Kolla, University of Colorado, Boulder; Lap Chi Lau, University of Waterloo; Jamie Morgenstern, University of Washington; Anand Natajaran, MIT; Alantha Newman, Université Grenoble Alpes; Lorenzo Orecchia, University of Chicago; Debmalya Panigrahi, Duke University; Richard Peng, Georgia Tech; Ron Rothblum, Technion; Aviad Rubinstein, Stanford; Tselil Schramm, Stanford; Leonard Schulman, California Institute of Technology; Anastasios Sidiropoulos, University of Illinois at Chicago; Nikhil Srivastava, UC Berkeley; Ola Svensson, EPFL; Avishay Tal, UC Berkeley; Luca Trevisan, Bocconi University; Jan Vondrak, Stanford; Matt Weinberg, Princeton; Amir Yehudayoff, Technion; Mark Zhandry, Princeton and NTT Research.

Planning and execution of the conference were made more challenging by a global pandemic, and in this light I am especially grateful to the committee for agreeing to serve, and then investing substantial time and effort in producing a fantastic program. Similarly, I am thankful to the hundreds of subreviewers who assisted in the reviewing process. Special thanks are also due to Gautam Kamath from the University of Waterloo, and the Simons Institute for the Theory of Computing for their help in organizing the virtual conference. I'm grateful to the ITCS Steering Committee – and especially its chair Umesh Vazirani – for their advice and guidance in forming the program and, finally, to all the authors, presenters, and audience members for making ITCS a unique and innovative experience.

James R. Lee  
ITCS 2021 Program Chair  
University of Washington  
Seattle, WA, USA





# ■ ITCS 2021 Conference Organization

**Program Chair:** James R. Lee (University of Washington)

**Technical Logistics:** Gautam Kamath (University of Waterloo)  
James R. Lee (University of Washington)

**Steering Committee Chair:** Umesh Vazirani (UC Berkeley)

**Steering Committee** Manuel Blum, Carnegie Mellon  
Bernard Chazelle, Princeton  
Irit Dinur, Weizmann  
Oded Goldreich, Weizmann  
Shafi Goldwasser, MIT and Weizmann  
Richard Karp, Berkeley  
Robert Kleinberg, Cornell University  
Ueli Maurer, ETH  
Silvio Micali, MIT  
Christos Papadimitriou, Berkeley  
Michael Rabin, Harvard  
Omer Reingold, Stanford  
Tim Roughgarden, Stanford  
Leslie Valiant, Harvard  
Umesh Vazirani, Berkeley  
Thomas Vidick, Caltech  
Avi Wigderson, IAS  
Andy Yao, Tsinghua

**Program Committee** Andris Ambainis, University of Latvia  
Nima Anari, Stanford  
Elette Boyle, IDC Herzliya  
Mark Braverman, Princeton  
Sebastien Bubeck, Microsoft Research  
Claire Mathieu, CNRS, Paris  
Edith Cohen, Google  
Anindya De, University of Pennsylvania  
Uriel Feige, Weizmann Institute  
Kira Goldner, Columbia  
Monika Henzinger, University of Vienna  
Maurice Herlihy, Brown  
Sam Hopkins, UC Berkeley and MIT  
Tali Kaufman, Bar-Ilan University  
Adam Klivans, UT Austin  
Gillat Kol, Princeton  
Alexandra Kolla, University of Colorado, Boulder  
Lap Chi Lau, University of Waterloo  
Anand Natajaran, MIT



<b>Program Committee</b>	Alantha Newman, Université Grenoble Alpes
<b>(continued)</b>	Lorenzo Orecchia, University of Chicago
	Debmalya Panigrahi, Duke University
	Richard Peng, Georgia Tech
	Ron Rothblum, Technion
	Aviad Rubinstein, Stanford
	Tselil Schramm, Stanford
	Leonard Schulman, California Institute of Technology
	Anastasios Sidiropoulos, University of Illinois at Chicago
	Nikhil Srivastava, UC Berkeley
	Ola Svensson, EPFL
	Avishay Tal, UC Berkeley
	Luca Trevisan, Bocconi University
	Jan Vondrak, Stanford
	Matt Weinberg, Princeton
	Amir Yehudayoff, Technion
	Mark Zhandry, Princeton and NTT Research

# The Entropy of Lies: Playing Twenty Questions with a Liar

**Yuval Dagan**

Massachusetts Institute of Technology, Cambridge, MA, USA  
dagan@mit.edu

**Yuval Filmus**

Technion – Israel Institute of Technology, Haifa, Israel  
yuvalfi@cs.technion.ac.il

**Daniel Kane**

University of California San Diego, La Jolla, CA, USA  
dakane@ucsd.edu

**Shay Moran**

Technion – Israel Institute of Technology, Haifa, Israel  
smoran@technion.ac.il

---

## Abstract

“Twenty questions” is a guessing game played by two players: Bob thinks of an integer between 1 and  $n$ , and Alice’s goal is to recover it using a minimal number of Yes/No questions. Shannon’s entropy has a natural interpretation in this context. It characterizes the average number of questions used by an optimal strategy in the distributional variant of the game: let  $\mu$  be a distribution over  $[n]$ , then the average number of questions used by an optimal strategy that recovers  $x \sim \mu$  is between  $H(\mu)$  and  $H(\mu) + 1$ .

We consider an extension of this game where at most  $k$  questions can be answered falsely. We extend the classical result by showing that an optimal strategy uses roughly  $H(\mu) + kH_2(\mu)$  questions, where  $H_2(\mu) = \sum_x \mu(x) \log \log \frac{1}{\mu(x)}$ . This also generalizes a result by Rivest et al. (1980) for the uniform distribution.

Moreover, we design near optimal strategies that only use comparison queries of the form “ $x \leq c$ ?” for  $c \in [n]$ . The usage of comparison queries lends itself naturally to the context of sorting, where we derive sorting algorithms in the presence of adversarial noise.

**2012 ACM Subject Classification** Theory of computation

**Keywords and phrases** entropy, twenty questions, algorithms, sorting

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.1

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/1811.02177>.

**Funding** *Yuval Filmus*: Taub Fellow – supported by the Taub Foundations. The research was funded by ISF grant 1337/16.

## 1 Introduction

The “twenty questions” game is a cooperative game between two players: Bob thinks of an integer between 1 and  $n$ , and Alice’s goal is to recover it using the minimal number of Yes/No questions. An optimal strategy for Alice is to perform binary search, using  $\log n$  queries in the worst case.

The game becomes more interesting when Bob chooses his number according to a distribution  $\mu$  known to both players, and Alice attempts to minimize the *expected* number of questions. In this case, the optimal strategy is to use a Huffman code for  $\mu$ , at an expected cost of roughly  $H(\mu)$ .



© Yuval Dagan, Yuval Filmus, Daniel Kane, and Shay Moran;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 1; pp. 1:1–1:16



Leibniz International Proceedings in Informatics

LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

What happens when Bob is allowed to lie (either out of spite, or due to difficulties in the communication channel)? Rényi [20] and Ulam [25] suggested a variant of the (non-distributional) “twenty questions” game, in which Bob is allowed to lie  $k$  times. Rivest et al. [21], using ideas of Berlekamp [4], showed that the optimal number of questions in this setting is roughly  $\log n + k \log \log n$ . There are many other ways of allowing Bob to lie, some of which are described by Spencer and Winkler [24] in their charming work, and many others by Pelc [19] in his comprehensive survey on the topic.

### Distributional “twenty questions” with lies

This work addresses the distributional “twenty questions” game in the presence of lies. In this setting, Bob draws an element  $x$  according to a distribution  $\mu$ , and Alice’s goal is to recover the element using as few Yes/No questions as possible on average. The twist is that Bob, who knows Alice’s strategy, is allowed to lie up to  $k$  times. Both Alice and Bob are allowed to use randomized strategies, and the average is measured according to both  $\mu$  and the randomness of both parties.

Our main result shows that the expected number of questions in this case is

$$H(\mu) + kH_2(\mu), \quad \text{where } H_2(\mu) = \sum_x \mu(x) \log \log \frac{1}{\mu(x)},$$

up to an additive factor of  $O(k \log k + kH_3(\mu))$ , where  $H_3(\mu) = \sum_x \mu(x) \log \log \log(1/\mu(x))$  (here  $\mu(x)$  is the probability of  $x$  under  $\mu$ .) See Section 3 for a complete statement of this result.

When  $\mu$  is the uniform distribution, the expected number of queries that our algorithm makes is roughly  $\log n + k \log \log n$ , matching the performance of the algorithm of Rivest et al. However, the approach by Rivest et al. is tailored to their setting, and the distributional setting requires new ideas.

As in the work of Rivest et al., our algorithms use only *comparison queries*, which are queries of the form “ $x \prec c$ ?” (for some fixed value  $c$ ). Moreover, our algorithms are efficient, requiring  $O(n)$  preprocessing time and  $O(\log n)$  time per question. Our lower bounds, in contrast, apply to *arbitrary* Yes/No queries.

### Noisy sorting

One can apply binary search algorithms to implement insertion sort. While sorting an array typically requires  $\Theta(n \log n)$  *sorting queries* of the form “ $x_i \prec x_j$ ?”, there are situations where one has some prior knowledge about the correct ordering. This may happen, for example, when maintaining a sorted array: one has to perform consecutive sorts, where each sort is not expected to considerably change the locations of the elements. Assuming a distribution  $\Pi$  over the  $n!$  possible permutations, Moran and Yehudayoff [17] showed that sorting a  $\Pi$ -distributed array requires  $H(\Pi) + O(n)$  sorting queries on average. We extend this result to the case in which the answerer is allowed to lie  $k$  times, giving an algorithm which uses the following expected number of queries:<sup>1</sup>

$$H(\Pi) + O(nk).$$

This result is tight, and matches the optimal algorithms for the uniform distribution due to Bagchi [3] and Long [16], which use  $n \log n + O(nk)$  queries.

---

<sup>1</sup> Strictly speaking, this bound holds only under the mild condition that  $k$  is at most exponential in  $n$ .

■ **Table 1** Query complexities of searching and sorting in different settings, ignoring lower-order terms. All terms are exact upper and lower bounds except for those inside the  $O(\cdot)$  and  $\Theta(\cdot)$  notations.

Setting	Searching	Sorting
No lies; deterministic	$\log n$ [classical]	$n \log n$ [classical]
No lies; distributional	$H(\mu)$ [classical]	$H(\Pi) + O(n)$ [17]
$k$ lies; deterministic	$\log n + k \log \log n$ [21]	$n \log n + \Theta(nk)$ [3, 16, 15]
$k$ lies; distributional	$H(\mu) + kH_2(\mu)$ [this paper]	$H(\Pi) + \Theta(nk)$ [this paper]

Table 1 summarizes the query complexities of resilient and non-resilient searching and sorting algorithms, in both the deterministic and the distributional settings. To the best of our knowledge, we present the first resilient algorithms in the distributional setting.

## On randomness

All algorithms presented in the paper are randomized. Since they only employ public randomness which is known for both players, there exists a fixing of the randomness which yields a deterministic algorithm with the same (or possibly smaller) expected number of queries. However, this comes at the cost of possibly increasing the running time of the algorithm (since we need to find a good fixing of the randomness); it would be interesting to derive an explicit efficient deterministic algorithm with a similar running time.

## 1.1 Main ideas

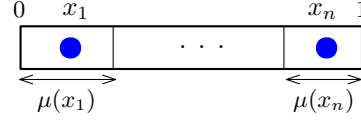
### Upper bound

Before presenting the ideas behind our algorithms, we explore several other ideas which give suboptimal results. The first approach that comes to mind is simulating the optimal non-resilient strategy, asking each question  $2k + 1$  times and taking the majority vote, which results in an algorithm using  $\Theta(kH(\mu))$  queries on average.

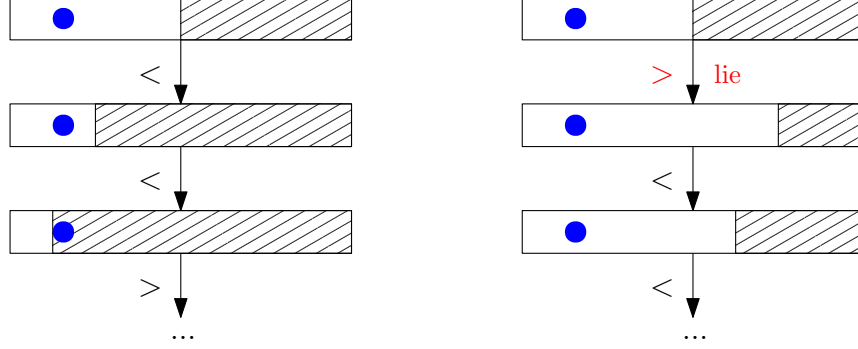
A better approach is using *tree codes*, suggested by Schulman [22] as an approach for making interactive communication resilient to errors [11, 22, 14]. Tree codes are designed for a different error model, in which we are bounding the *fraction* of lies rather than their absolute number; for an  $\varepsilon$ -fraction of lies, the best known constructions suffer a multiplicative overhead of  $1 + O(\sqrt{\varepsilon})$  [13]. In contrast, we are aiming at an *additive* overhead of  $kH_2(\mu)$ .

Using a packing bound, one can prove that there exists a (non-interactive) code of expected length roughly  $H(\mu) + 2kH_2(\mu)$ , coming much closer to the bound that we are able to get (but off by a factor of 2 from our target  $H(\mu) + kH_2(\mu)$ ). The idea, which is similar to the proof of the Gilbert–Varshamov bound, is to construct a prefix code  $w_1, \dots, w_n$  in which the prefixes of  $w_i, w_j$  of length  $\min(|w_i|, |w_j|)$  are at distance at least  $2k + 1$  (whence the factor  $2k$  in the resulting bound); this can be done greedily. Apart from the inferior bound, two other disadvantages of this approach is that it is not efficient and uses arbitrary queries.

In contrast to these prior techniques, which do not achieve the optimal complexity, might ask arbitrary questions, and could result in strategies which cannot be implemented efficiently, in this paper we design an efficient and nearly optimal strategy, relying on comparison queries only, and utilizing simple observations on the behavior of binary search trees under the presence of lies.



■ **Figure 1** Representing items as centers of segments partitioning the interval  $[0, 1]$ .



■ **Figure 2** On the left, the operation of the algorithm without any lies. On the right, answerer lied on the first question. As a result, all future truthful answers are the same.

Following the footsteps of Rivest et al. [21], our upper bound is based on a binary search algorithm on the unit interval  $[0, 1]$ , first suggested in this context by Gilbert and Moore [12]: given  $x \in [0, 1]$ , the algorithm locates  $x$  by first asking “ $x < 1/2$ ?”; depending on the answer, asking “ $x < 1/4$ ?” or “ $x < 3/4$ ?”; and so on. If  $x \in [0, 1]$  is chosen uniformly at random then the answers behave like an infinite sequence of random and uniform coin tosses.

In order to apply this kind of binary search to the problem of identifying an unknown element (assuming truthful answers), we partition the unit interval  $[0, 1]$  into segments of lengths  $\mu(x_1), \dots, \mu(x_n)$ , and label the center of each segment with the corresponding item (see Figure 1). We then perform binary search until the current interval contains a single item. (In the proof, we use a slightly more sophisticated randomized placement of points which guarantees that the answers on *each* element behave like an infinite sequence of random and uniform coin tosses.)

The main observation is that if a question “ $x < a$ ?” is answered with a lie, this will be strongly reflected in subsequent answers (see Figure 2). Indeed, suppose that  $x < a$ , but Bob claimed that  $x > a$ . All subsequent questions will be of the form “ $x < b$ ?” for various  $b > a$ , the truthful answer to all of which is  $x < b$ . An observer taking notes of the proceedings will thus observe the following pattern of answers:  $>$  (the lie) followed by many  $<$ ’s (possibly interspersed with up to  $k - 1$  many  $>$ ’s, due to further lies). This is suspicious since it is highly unlikely to obtain many  $<$  answers in a row (the probability of getting  $r$  such answers is just  $2^{-r}$ ).

This suggests the following approach: for each question we will maintain a “confidence interval” consisting of  $r(d)$  further questions (where  $d$  is the index of the question). At the end of the interval, we will check whether the situation is suspicious (as described in the preceding paragraph), and if so, will ascertain by brute force the correct answer to the original question (by taking a majority of  $2k + 1$  answers), and restart the algorithm from that point.



The best choice for  $r(d)$  turns out to be roughly  $\log d$ . Each time Bob lies, our unrolling of the confidence interval results in a loss of  $r(d)$  questions. Since an item  $x$  requires roughly  $\log(1/\mu(x))$  questions to be discovered, the algorithm has an overhead of roughly  $kr(\log(1/\mu(x))) \approx k \log \log(1/\mu(x))$  questions on element  $x$ , resulting in an expected overhead of roughly  $kH_2(\mu)$ .

When implementing the algorithm, apart from the initial  $O(n)$  time needed to setup the partition of  $[0, 1]$  into segments, the costliest step is to convert the intervals encountered in the binary search to comparison queries. This can be done in  $O(\log n)$  time per query.

## Lower bound

The proof of our lower bound uses information theory: one can lower bound the expected number of questions by the amount of information that the questioner gains. There are two such types of information: first, the hidden object reveals  $H(\mu)$  information, as in the setting where no lies are allowed. Second, when the object is revealed, the positions of the lies are revealed as well. This reveals additional  $H_2(\mu)$  (conditional) information, as we explain below.

Let  $d_x$  denote the number of questions asked for element  $x$ . Kraft's inequality shows that any good strategy of the questioner satisfies  $d_x \gtrsim \log(1/\mu(x))$ . If the answerer chooses a randomized strategy in which the positions of the lies are chosen uniformly from the  $\binom{d_x}{k}$  possibilities, these positions reveal  $\log \binom{d_x}{k} \approx k \log d_x \gtrsim k \log \log(1/\mu(x))$  information given  $x$ . Taking expectation over  $x$ , the positions of the lies reveal at least  $kH_2(\mu)$  information beyond the identity of  $x$ .

## 1.2 Related work

Most of the literature on error-resilient search procedures has concentrated on the non-distributional setting, in which the goal is to give a worst case guarantee on the number of questions asked, under various error models. The most common error models are as follows:<sup>2</sup>

- Fixed number of errors. This is the error model we consider, and it is also the one suggested by Ulam [25]. This model was first studied by Berlekamp [4], who used an argument similar to the sphere-packing bound to give a lower bound on the number of questions. Rivest et al. [21] used this lower bound as a guiding principle in their almost matching upper bound using comparison queries.
- At most a fixed fraction  $p$  of the answers can be lies. This model is similar to the one considered in error-correcting codes. Pelc [18] and Spencer and Winkler [24] (independently) gave a non-adaptive strategy for revealing the hidden element when  $p \leq 1/4$ , and showed that the task is not possible (non-adaptively) when  $p > 1/4$ . Furthermore, when  $p < 1/4$  there is an algorithm using  $O(\log n)$  questions, and when  $p = 1/4$  there is an algorithm using  $O(n)$  questions, which are both optimal (up to constant factors). Spencer and Winkler also showed that if questions are allowed to be adaptive, then the hidden element can be revealed if and only if  $p < 1/3$ , again using  $O(\log n)$  questions.
- At most a fixed fraction  $p$  of any *prefix* of the answers can be lies. Pelc [18] showed that the hidden element can be revealed if and only if  $p < 1/2$ , and gave an  $O(\log n)$  strategy when  $p < 1/4$ . Aslam and Dhagat [2] and Spencer and Winkler gave an  $O(\log n)$  strategy for all  $p < 1/2$ .

<sup>2</sup> This section is heavily based on Pelc's excellent and comprehensive survey [19].

- Every question is answered erroneously with probability  $p$ , an error model common in information theory. Rényi [20] showed that the number of questions required to discover the hidden element with constant success probability is  $(1 + o(1)) \log n / (1 - h(p))$ .

The distributional version of the “twenty questions” game (without lies) was first considered by Shannon [23] in his seminal paper introducing information theory, where its solution was attributed to Fano (who published it later as [9]). The Shannon–Fano code uses at most  $H(\mu) + 1$  questions on average, but the questions can be arbitrary. The Shannon–Fano–Elias code (also due to Gilbert and Moore [12]), which uses only comparison queries, asks at most  $H(\mu) + 2$  questions on average. Dagan et al. [7] give a strategy, using only comparison and equality queries, which asks at most  $H(\mu) + 1$  questions on average.

### Sorting

The non-distributional version of sorting has also been considered in some of the settings considered above:

- At most  $k$  errors: Lakshmanan et al. [15] gave a lower bound of  $\Omega(n \log n + kn)$  on the number of questions, and an almost matching upper bound of  $O(n \log n + kn + k^2)$  questions. An optimal algorithm, using  $n \log n + O(kn)$  questions, was given independently by Bagchi [3] and Long [16].
- At most a  $p$  fraction of errors in every prefix: Aigner [1] showed that sorting is possible if and only if  $p < 1/2$ . Borgstrom and Kosaraju [5] had showed earlier that even *verifying* that an array is sorted requires  $p < 1/2$ .
- Every answer is correct with probability  $p$ : Feige et al. [10] showed in an influential paper that  $\Theta(n \log(n/\epsilon))$  queries are needed, where  $\epsilon$  is the probability of error.
- Braverman and Mossel [6] considered a different setting, in which an algorithm is given access to noisy answers to all possible  $\binom{n}{2}$  comparisons, and the goal is to find the most likely permutation. They gave a polynomial time algorithm which succeeds with high probability.

The distributional version of sorting (without lies) was considered by Moran and Yehudayoff [17], who gave a strategy using at most  $H(\mu) + 2n$  queries on average, based on the Gilbert–Moore algorithm.

### Paper organization

After a few preliminaries in Section 2, we describe our results in full in Section 3. The proofs are presented in the full version of this paper [8].

## 2 Definitions

We use the notation  $\binom{n}{\leq k} = \sum_{\ell=0}^k \binom{n}{\ell}$ . Unless stated otherwise, all logarithms are base 2. We define  $\overline{\log}(x) = \log(x + C)$  and  $\overline{\ln}(x) = \ln(x + C)$  for a fixed sufficiently large constant  $C > 0$  satisfying  $\log \log \log C > 0$ .

### Information theory

Given a probability distribution  $\mu$  with countable support, the entropy of  $\mu$  is given by the formula

$$H(\mu) = \sum_{x \in \text{supp } \mu} \mu(x) \log \frac{1}{\mu(x)}.$$

## Twenty questions game

We start with an intuitive definition of the game, played by a questioner (Alice) and an answerer (Bob). Let  $U$  be a finite set of elements, and let  $\mu$  be a distribution over  $U$ , known to both parties. The game proceeds as follows: first, an element  $x \sim \mu$  is drawn and revealed to the answerer but not to the questioner. The element  $x$  is called the *hidden* element. The questioner asks binary queries of the form “ $x \in Q?$ ” for subsets  $Q \subseteq U$ . The answerer is allowed to lie a fixed number of times, and the goal of the questioner is to recover the hidden element  $x$ , asking the minimal number of questions on expectation.

## Decision trees

Let  $U$  be a finite set of elements. A *decision tree*  $T$  for  $U$  is a binary tree formalizing the question asking strategy in the twenty questions game. Each internal node of  $T$  is labeled by a *query* (or *question*) – a subset of  $U$ , denoted by  $Q(v)$ ; and each leaf is labeled by the output of the decision tree, which is an element of  $U$ . The semantics of the tree are as follows: on input  $x \in U$ , traverse the tree by starting at the root, and whenever at an internal node  $v$ , go to the left child if  $x \in Q(v)$  and to the right child if  $x \notin Q(v)$ .

## Comparison tree

Given an ordered set of elements  $x_1 \prec x_2 \prec \dots \prec x_n$ , *comparison questions* are questions of the form  $Q = \{x_1, \dots, x_{i-1}\}$ , for some  $i = 1, \dots, n+1$ . In other words, the questions are “ $x \prec x_i?$ ” for some  $i = 1, \dots, n+1$ . An answer to a comparison question is one of  $\{\prec, \succeq\}$ . A *comparison tree* is a decision tree all of whose nodes are labeled by comparison questions.

## Adversaries

Let  $k \geq 0$  be a bound on the number of lies. An intuitive way to formalize the possibility of lying is via an adversary. The adversary knows the hidden element  $x$  and receives the queries from the questioner as the tree is being traversed. The adversary is allowed to lie at most  $k$  times, where each lie is a violation of the above stated rule. Formally, an adversary is a mapping that receives as input an element  $x \in X$ , a sequence of the previous queries and their answers, and an additional query  $Q \subseteq U$ , which represents the current query. The output of the adversary is a boolean answer to the current query; this answer is a *lie* if it differs from the truth value of “ $x \in Q$ ”.

We also allow the adversary and the tree to use randomness: a randomized decision tree is a distribution over decision trees and a randomized adversary is a distribution over adversaries.

## Computation and complexity

The responses of the adversary induce a unique root-to-leaf path in the decision tree, which results in the output of the tree. A decision tree is *k-valid* if it outputs the correct element against any adversary that lies at most  $k$  times.

Given a  $k$ -valid decision tree  $T$  and a distribution  $\mu$  on  $U$ , the *cost* of  $T$  with respect to  $\mu$ , denoted  $c(T, \mu)$ , is the maximum, over all possible adversaries that lie at most  $k$  times, of the expected<sup>3</sup> length of the induced root-to-leaf path in  $T$ . Finally, the *k-cost* of  $\mu$ , denoted  $c_k(\mu)$ , is the minimum of  $c(T, \mu)$  over all  $k$ -valid decision trees  $T$ .

<sup>3</sup> The expectation is also taken with respect to the randomness of the adversary and the tree when they are randomized.

### Basic facts

We will refer to the following well-known formula as *Kraft's identity*:

► **Fact 1** (Kraft's identity). *Fix a binary tree  $T$ , let  $L$  be its set of leaves and let  $d(\ell)$  be the depth of leaf  $\ell$ . The following applies:*

$$\sum_{\ell \in L(T)} 2^{-d(\ell)} \leq 1.$$

We will use the following basic lower bound on the expected depth by the entropy:

► **Fact 2.** *Let  $T$  be a binary tree and let  $\mu$  be a distribution over its leaves. Then*

$$H(\mu) \leq \mathbb{E}_{\ell \sim \mu} [d(\ell)].$$

In other words, for any distribution  $\mu$ ,  $c_0(\mu) \geq H(\mu)$ . In fact, it is also known that  $c_0(\mu) \leq H(\mu) + 1$ .

## 3 Main results

This section is organized as follows: The lower bound is presented in Section 3.1. Then, the two searching algorithms are presented in Section 3.2, and finally the application to sorting is presented in Section 3.3.

### 3.1 Lower bound

In this section we present the following lower bound on  $c_k(\mu)$ , namely, on the expected number of questions asked by *any*  $k$ -valid tree (not necessarily a comparison trees).

► **Theorem 3.** *For every non-constant distribution  $\mu$  and every  $k \geq 0$ ,*

$$c_k(\mu) \geq \left( \mathbb{E}_{x \sim \mu} \log \frac{1}{\mu(x)} \right) + k \left( \mathbb{E}_{x \sim \mu} \log \log \frac{1}{\mu(x)} \right) - (k \log k + k + 1).$$

### Proof overview

Consider a  $k$ -valid tree; we wish to lower bound the expected number of questions for  $x \sim \mu$ . Let  $d_x$  denote the number of questions asked when the secret element is  $x$ . Then, by the entropy lower bound when the number of mistakes is  $k = 0$ , it follows that *typically*,  $d_x \gtrsim \log(1/\mu(x))$ . Moreover, the transcript of the game (i.e. the list of questions and answers) determines both  $x$  and the positions of the  $k$  lies. This requires

$$d_x + k \log(d_x) \gtrsim \log(1/\mu(x)) + k \log \log(1/\mu(x))$$

bits of information. Taking expectation over  $x \sim \mu$  then yields the stated bound.

Our proof formalizes this intuition using standard and basic tools from information theory. One part that requires a subtler argument is showing that indeed one may assume that  $d_x \gtrsim \log(1/\mu(x))$  for all  $x$ . This is done by showing that any  $k$ -valid tree can be modified to satisfy this constraint without increasing the expected number of questions by too much.

### 3.2 Upper bounds

We introduce two algorithms. The first algorithm, presented in Section 3.2.1, is simpler, however, the second algorithm has a better query complexity. The expected number of questions asked by the first algorithm is at most

$$H(\mu) + (k+1)H_2(\mu) + O(k^2 H_3(\mu) + k^2 \log k), \quad \text{where } H_3(\mu) = \sum_x \mu(x) \log \log \log \frac{1}{\mu(x)}.$$

The second algorithm, presented in Section 3.2.2, removes the quadratic dependence on  $k$ , and has an expected complexity of:

$$H(\mu) + kH_2(\mu) + O(kH_3(\mu) + k \log k).$$

In Section 3.2.3 we robustify the guarantees of these algorithms and consider scenarios where the exact distribution  $\mu$  is not known but only some prior  $\eta \approx \mu$ , or where the actual number of lies is less than the bound  $k$  (whence the algorithm achieves better performance).

#### 3.2.1 First algorithm

Suppose that we are given a probability distribution  $\mu$  whose support is the linearly ordered set  $x_1 \prec \dots \prec x_n$ . In this section we overview the proof of the following theorem:

► **Theorem 4.** *There is a  $k$ -valid comparison tree  $T$  with*

$$c(T, \mu) \leq H(\mu) + (k+1) \sum_{i=1}^n \mu_i \log \log \frac{1}{\mu_i} + O\left(k^2 \sum_{i=1}^n \mu_i \log \log \log \frac{1}{\mu_i} + k^2 \log k\right),$$

where  $\mu_i = \mu(x_i)$ .

The question-asking strategy simulates a binary search to recover the hidden element. If, at some point, the answer to some question  $q$  is suspected as a lie then  $q$  is asked  $2k+1$  times to verify its answer. When is the answer to  $q$  suspected? The binary search tree is constructed in a manner that if no lies are told then roughly half of the questions are answered  $\prec$ , and half  $\succeq$ . However, if, for example, the lie “ $x \succeq x_{50}$ ” is told when in fact  $x = x_{10}$ , then all consecutive questions will be of the form “ $x \prec x_i$ ?” for  $i > 50$ , and the correct answer would always be  $\prec$ . Since no more than  $k$  lies can be told, almost all consecutive questions will be answered  $\prec$ , and the algorithm will suspect that some earlier question is a lie.

We start by suggesting a question-asking strategy using comparison queries which is valid as long as there are no lies, and then show how to make it resilient to lies. Each element  $x_i$  is mapped to a point  $p_i$  in  $[0, 1]$ , such that  $p_1 < p_2 < \dots < p_n$ . Then, a binary search on the interval  $[0, 1]$  is performed, for finding the point  $p_i$  corresponding to the hidden element. The search proceeds by maintaining a *Live* interval, which is initialized to  $[0, 1]$ . At any iteration, the questioner asks whether  $p_i$  lies in the left half of the *Live* interval. The interval is updated accordingly, and its length shrinks by a factor of 2. This technique was proposed by Gilbert–Moore [12], and is presented in AuxiliaryAlgorithm 1, as an algorithm which keeps asking questions indefinitely.

The points  $p_1, \dots, p_n$  are defined as follows: first, a number  $\theta \in [0, 1/2)$  is drawn uniformly at random. Now, for any element  $i$  define  $p_i = \frac{1}{2} \sum_{j=1}^{i-1} \mu_j + \frac{1}{4} \mu_i + \theta$ .<sup>4</sup> Given  $\theta$ , let  $T'_\theta$  denote the infinite tree generated by AuxiliaryAlgorithm 1. Note that whenever *Live* contains just

<sup>4</sup> In the original paper  $p_i$  was defined similarly but without the randomization:  $p_i = \sum_{j=1}^{i-1} \mu_j + \frac{1}{2} \mu_i$ .

---

**Algorithm 1** Randomized Gilbert–Moore.

---

```

1:  $Live \leftarrow [0, 1]$ 
2: loop
3:    $m \leftarrow \text{midpoint of } Live$ 
4:    $X \leftarrow \{i : p_i \geq m\}$ 
5:   if  $x \in X$  then
6:      $Live \leftarrow \text{right half of } Live$ 
7:   else
8:      $Live \leftarrow \text{left half of } Live$ 
9:   end if
10: end loop

```

---

one point  $p_i$ , then (as there are no lies) the hidden element must be  $x_i$ . Denote by  $T_\theta$  the finite tree corresponding to the algorithm which stops whenever that happens. We present two claims about these trees.

First, conditioned on any hidden element  $x_i$ , the answers to all questions (except, perhaps, for the first answer) are distributed uniformly and independently, where the distribution is over the random choice of  $\theta$ . This follows from the fact that all bits of  $p_i$  except for the most significant bit are i.i.d. unbiased coin flips.

▷ **Claim 5.** For any element  $x_i$ , let  $(A_t)$  be the random sequence of answers to the questions in AuxiliaryAlgorithm 1, containing all answers except for the first answer, assuming there are no lies. The distribution of the sequence  $(A_t)$  is the same as that of an infinite sequence of independent unbiased coin tosses, where the randomness stems from the random choice of  $p_i$ .

Second, since  $\min(p_i - p_{i-1}, p_{i+1} - p_i) \geq \mu_i/4$ , one can bound the time it takes to isolate  $x_i$  as follows.

▷ **Claim 6.** For any element  $x_i$  and any  $\theta$ , the leaf in  $T_\theta$  labeled by  $x_i$  is of depth at most  $\log(1/\mu_i) + 3$ . Hence, if  $x$  is drawn from a distribution  $\mu$ , the expected depth of the leaf labeled  $x$  is at most  $\sum_i \mu_i \log(1/\mu_i) + 3 = H(\mu) + 3$ .

We now describe the  $k$ -resilient algorithm: Algorithm 1 (the pseudocode appears as well). At the beginning, a number  $\theta$  is randomly drawn. Then, two concurrent simulations over  $T'_\theta$  are performed, and two pointers to nodes in this tree are maintained (recall that  $T'_\theta$  is the infinite binary search tree). The first pointer, *Current*, simulates the question-asking strategy according to  $T'_\theta$ , ignoring the possibility of lies. In particular, it may point on an incorrect node in the tree (reached as a result of a lie). Since *Current* ignores the possibility of lies, there is a different pointer, *LastVerified*, which verifies the answers to the questions asked in the simulation of *Current*. All answers in the path from the root to *LastVerified* are verified as correct, and *LastVerified* will always be an ancestor of *Current*. See Figure 3 for the basic setup.

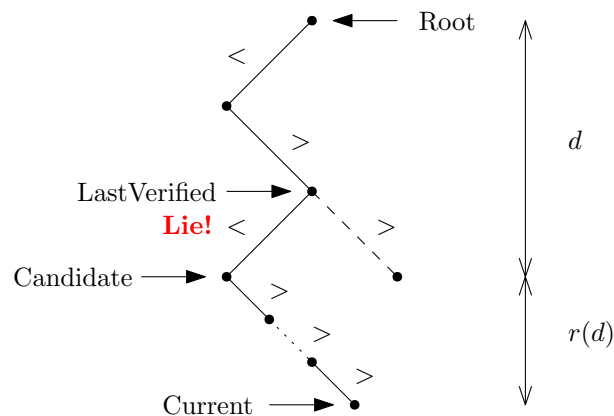
The algorithm proceeds in iterations. In every iteration the question  $Q(\textit{Current})$  is asked and *Current* is advanced to the corresponding child accordingly. In some of the iterations also *LastVerified* is advanced. Concretely, this happens when the depth of *Current* in  $T'_\theta$  equals  $d + r(d)$ , where  $d$  is the depth of *LastVerified* and  $r(d) \approx \log d + k \log \log d$ . In these iterations, the answer given to  $Q(\textit{LastVerified})$  is being verified, as detailed next.

■ **Algorithm 1** Resilient-Tree.

```

1:  $\theta \leftarrow \text{Uniform}([0, 1/2])$ 
2:  $Current \leftarrow \text{root}(T'_\theta)$ 
3:  $LastVerified \leftarrow \text{root}(T'_\theta)$ 
4: while  $LastVerified$  is not a leaf of  $T_\theta$  do
5:   if  $x \in Q(Current)$  then
6:      $Current \leftarrow \text{left-child}(Current)$ 
7:   else
8:      $Current \leftarrow \text{right-child}(Current)$ 
9:   end if
10:   $d \leftarrow \text{depth}(LastVerified) + 1$ 
11:  if  $\text{depth}(Current) = d + r(d)$  then
12:     $Candidate \leftarrow$  child of  $LastVerified$  which is an ancestor of  $Current$ 
13:     $VerificationPath \leftarrow$  ancestors of  $Current$  up to and excluding  $Candidate$ 
14:    if  $Candidate$  is a left (right) child and at most  $k - 1$  vertices in  $VerificationPath$ 
are left (right) children then
15:      Ask  $2k + 1$  times the question  $x \in Q(LastVerified)$ 
16:      if majority answer is  $x \in Q(LastVerified)$  then
17:         $LastVerified \leftarrow \text{left-child}(LastVerified)$ 
18:      else
19:         $LastVerified \leftarrow \text{right-child}(LastVerified)$ 
20:      end if
21:       $Current \leftarrow LastVerified$ 
22:    else
23:       $LastVerified \leftarrow Candidate$ 
24:    end if
25:  end if
26: end while
27: return label of  $LastVerified$ 

```



■ **Figure 3** An illustration of Algorithm 1 just before the detection of a lie. The answer at *LastVerified* was a lie ( $<$  instead of  $>$ ), and so all answers below *Candidate* (except for any further lies) are  $>$ . This is noticed since *Current* is at depth  $d + r(d)$ . The answer at *Candidate* will be verified and found wrong, and so *LastVerified* would move to the sibling of *Candidate* (and so will *Current*), and the algorithm will continue from that point.

### The verification process

Next, we examine the verification process when *LastVerified* is advanced. There are two possibilities: first, when the answer to the question  $Q(\textit{LastVerified})$ , which was given when *Current* traversed it, is verified to be correct. In that case, *LastVerified* moves to its child which lies on the path towards *Current*. In the complementing case, when the the answer to the question  $Q(\textit{LastVerified})$  is detected as a lie, then *LastVerified* moves to the other child. In that case, *Current* is no longer a descendant of *LastVerified*, hence *Current* is moved up the tree and is set to *LastVerified*.

We now explain how the answer to  $Q(\textit{LastVerified})$  is verified. There are two verification steps: the first step uses no additional questions and the second step uses  $2k + 1$  additional questions. Usually, only the first step will be used and no additional questions will be spent during verification. In the first verification step one checks whether the following condition holds:

*The answer to  $Q(\textit{LastVerified})$  is identical to at least  $k$  of the answers along the path from  $\textit{LastVerified}$  to  $\textit{Current}$ .*

If this condition holds, then the answer is verified as correct. To see why this reasoning is valid, assume without loss of generality that the answer is  $\prec$ , and assume towards contradiction that it was a lie. Then, the correct answers to all following questions in the simulation of *Current* are  $\succeq$ . Since there can be at most  $k - 1$  additional lies, there can be at most  $k - 1$  additional  $\prec$  answers. Hence, if there are more  $\prec$  answers among the following questions then the previous answer to  $Q(\textit{LastVerified})$  is verified as correct.

Else, if the above condition does not hold then one proceeds to the second verification step and asks  $2k + 1$  times the question  $Q(\textit{LastVerified})$ . Here, the majority answer must be correct, since there can be at most  $k$  lies.

We add one comment: if the second verification step is taken, one sets  $\textit{Current} \leftarrow \textit{LastVerified}$  regardless of whether a lie had been revealed (this is performed to facilitate the proof). So, whenever the condition in the first verification step fails to hold then *Current* and *LastVerified* point to the same node in the tree.

The algorithm ends when *LastVerified* reaches a leaf of  $T_\theta$ , at which point the hidden element is recovered.

### Query complexity analysis

Fix an element  $x_i$ . We bound the expected number of questions asked when  $x_i$  is the hidden element as follows. Define  $d \approx \log(1/\mu(x_i))$  as the depth of the leaf labeled  $x_i$  in  $T_\theta$ . We divide the questions into the following five categories:

- Questions on the path  $P$  from the root to *Current* by the end of the algorithm, when *Current* reaches depth  $d + r(d)$ , *LastVerified* reaches depth  $d$ , and the algorithm terminates. Hence, there are at most  $d + r(d)$  such questions.
- Questions that were ignored due to the second verification step while *Current* was backtracked from a node outside  $P$ . This can only happen due to a lie between *Current* and *LastVerified* so there are at most  $k \cdot r(d)$  such questions.
- Questions asked  $2k + 1$  times during the second verification step when *Current* was pointing to a node outside  $P$ . This can only happen due to a lie between *Current* and *LastVerified* so there are at most  $k \cdot (2k + 1)$  such questions.
- Questions that were ignored due to the second verification step, when *Current* was being backtracked from a node in  $P$ . By the choice of  $r(d)$  there are at most  $O(1)$  such questions (on expectation).
- Questions asked  $2k + 1$  during the second verification step when *Current* was pointing to a node in  $P$ . By the choice of  $r(d)$  there are at most  $O(1)$  such questions (in expectation).



Summing these bounds up, one obtains a bound of

$$\begin{aligned} & (d + r(d)) + k \cdot r(d) + k \cdot (2k + 1) + O(1) + O(1) \\ & \approx \log(1/\mu_i) + (k + 1) \left( \log \log(1/\mu_i) + k \log \log \log(1/\mu_i) + O(k) \right). \end{aligned}$$

### 3.2.2 Second algorithm

In this section we overview the proof of the following theorem.

► **Theorem 7.** *For any distribution  $\mu$  there exists a  $k$ -valid comparison tree  $T$  with*

$$c(T, \mu) \leq H(\mu) + k \mathbb{E}_{x \sim \mu} [\log \log(1/\mu(x))] + O(k \mathbb{E}_{x \sim \mu} [\log \log \log(1/\mu(x))] + k \log k).$$

We explain the key differences with Algorithm 1.

- In Algorithm 1, an answer to a question  $Q$  at depth  $d$  was suspected as a lie if at most  $k$  of the  $r(d)$  consecutive questions received the same answer as  $Q$ . In the new algorithm, we suspect a question  $Q$  if *all* the  $r'(d)$  consecutive answers are different than  $Q$ . This change enables setting  $r'(d) \approx \log d$  rather than the previous value of  $r(d) \approx \log d + k \log \log d$ . Similarly to Algorithm 1, any time a lie is deleted,  $r'(d)$  questions are being deleted. Summing over the  $k$  lies, one obtains a total of  $kr'(d) \approx k \log d$  deleted questions, which is smaller than the corresponding value of  $kr(d) \approx k \log d + k^2 \log \log d$  in Algorithm 1.
- In Algorithm 1, the lies were detected in the same order they were told (i.e. in a *first-in-first-out* queue-like manner). This is due to the semantic of the pointer *LastVerified* which verifies the questions one-by-one, along the branching of the tree. In Algorithm 2 the pointer *LastVerified* is removed (only *Current* is used), and the lies are detected in a *last-in-first-out* stack-like manner: only the last lie can be deleted at any point in time. Indeed, as described in the previous paragraph, a lie will be deleted only if all consecutive answers are different (which is equivalent to them being non-lies).
- In Algorithm 1, when an answer is suspected as a lie, the corresponding question  $Q$  is repeated  $2k + 1$  times in order to verify its correctness. This happens after each lie, hence  $\Omega(k)$  redundant questions are asked per lie. In Algorithm 2, the suspected question  $Q$  will be asked again only *once*, and the algorithm will proceed accordingly. It may however be the case that this process will repeat itself and also the second answer to this question will be suspected as a lie and  $Q$  will be asked once again and so on. In order to avoid an infinite loop we add the condition that if the same answer is told  $k + 1$  times then it is guaranteed to be correct and will not be suspected any more.
- The removal of *LastVerified* forces finding a different method of verifying the correctness of an element  $x$  upon arriving at a leaf of  $T_\theta$ . One option is to ask the question “element =  $x$ ?”  $2k + 1$  times and take the majority vote, where each = question is implemented using one  $\preceq$  and one  $\succeq$ . This will, however, lead to asking  $\Omega(k)$  redundant questions each time  $x$  is not the correct element. Instead, one asks “element =  $x$ ?” multiple times, stopping either when the answer = is obtained  $k + 1$  times, or by the first the answer  $\neq$  has obtained more than the answer =. The total redundancy imposed by these verification questions throughout the whole search is  $O(k)$ .

To put the algorithm together, we exploit some simple combinatorial properties of paths containing multiple lies.

### 3.2.3 A fine-grained analysis of the guarantees

In this section, we present a stronger statement for the guarantees of our algorithms. First, the algorithms do not have to know *exactly* the distribution  $\mu$  from which the hidden element is drawn: an approximation suffices for getting a similar bound. Recall that the algorithm gets as an input some probability distribution  $\eta$ . This distribution might differ from the true distribution  $\mu$ . The cost of using  $\eta$  rather than  $\mu$  is related to  $D(\mu\|\eta)$ , the Kullback–Leibler divergence between the distributions.

Secondly, the algorithm has stronger guarantees when the actual number of lies is less than  $k$ . This is an improvement comparing to the algorithm of Rivest et al. [21] mentioned in the introduction. It will be utilized in the application of sorting, where the searching algorithm is invoked multiple times with a bound on the total number of lies (rather the number of lies per iteration). We present the general statement with respect to Algorithm 2.

► **Theorem 8.** *Assume that Algorithm 2 is invoked with the distribution  $(\eta_1, \dots, \eta_n)$ . Then, for any element  $x_i$ , the expected number of questions asked when  $x_i$  is the secret is at most*

$$\log(1/\eta_i) + \mathbb{E}[K'] \log \log \frac{1}{\eta_i} + O\left(\mathbb{E}[K'] \log \log \log \frac{1}{\eta_i} + \mathbb{E}[K'] \log k + k\right),$$

where  $K'$  is the expected number of lies. (The expectation is taken over the randomness of both parties.)

As a corollary, one obtains Theorem 7 and the following corollary, which corresponds to using a distribution different from the actual distribution.

► **Corollary 9.** *Assume that Algorithm 2 is invoked with  $(\eta_1, \dots, \eta_n)$  while  $(\mu_1, \dots, \mu_n)$  is the true distribution. Then, for a random hidden element drawn from  $\mu$ , the expected number of questions asked is at most*

$$\begin{aligned} H(\mu) + k \mathbb{E}_{x \sim \mu} [\log \log(1/\mu(x))] + O(k \mathbb{E}_{\mu} [\log \log \log(1/\mu(x))] + k \log k) \\ + D(\mu\|\eta) + O(k \log D(\mu\|\eta)), \end{aligned}$$

where  $D(\mu\|\eta) = \sum_{x \in \text{supp } \mu} \mu(x) \log \frac{\mu(x)}{\eta(x)}$  is the Kullback–Leibler divergence between  $\mu$  and  $\eta$ .

Corollary 9 follows from Theorem 8 by bounding  $K' \leq k$ , taking expectation over  $x_i \sim \mu$ , noting that  $\sum_i \mu_i \log(1/\eta_i) = H(\mu) + D(\mu\|\eta)$  and applying Jensen's inequality with the function  $x \mapsto \log x$ .

## 3.3 Sorting

One can apply Algorithm 2 to implement a stable version of the insertion sort using comparison queries. Let  $\Pi$  be a distribution over the set of permutations on  $n$  elements. Complementing with prior algorithms achieving a complexity of  $H(\Pi) + O(n)$  in the randomized setting with no lies [17], and  $n \log n + O(nk + n)$  in the deterministic setting with  $k$  lies [3, 16], we present an algorithm with a complexity of  $H(\Pi) + O(nk + n + k \log k)$  in the distributed setting with  $k$  lies. Note that  $k \log k = O(nk)$  unless the unlikely case that  $k = e^{w(n)}$ , hence the  $k \log k$  term can be ignored. Therefore, the guarantee of our algorithm matches the guarantees of the prior algorithms substituting either  $k = 0$  or  $\Pi = \text{Uniform}$ .

► **Theorem 10.** *Assume a distribution  $\Pi$  over the set of all permutations on  $n$  elements. There exists a sorting algorithm which is resistant to  $k$  lies and sorts the elements using  $H(\Pi) + O(nk + n + k \log k)$  comparisons on expectation.*

The randomized algorithms benefit from prior knowledge, namely, when one has information about the correct ordering. This is especially useful for maintaining a sorted list of elements, a procedure common in many sequential algorithms. In these settings, the values of the elements can change in time, hence, the elements have to be re-sorted regularly, however, their locations are not expected to change drastically.

The suggested sorting algorithm performs  $n$  iterations of insertion sort. By the end of each iteration  $i$ ,  $x_1, \dots, x_i$  are successfully sorted. Then, on iteration  $i + 1$ , one performs a binary search to find the location where  $x_{i+1}$  should be inserted, using conditional probabilities.

The guarantee of the algorithm is asymptotically tight: a lower bound of  $H(\Pi)$  follows from information theoretic reasons, and a lower bound of  $\Omega(nk)$  follows as well: the bound of Lakshmanan et al. [15] can be adjusted to the randomized setting.

---

## References

- 1 Martin Aigner. Finding the maximum and minimum. *Discrete Applied Mathematics*, 74:1–12, 1997.
- 2 Javed A. Aslam and Aditi Dhagat. Searching in the presence of linearly bounded errors. In *Proceedings of the 23rd annual Symposium on Theory of Computing (STOC'91)*, pages 486–493, 1991.
- 3 A. Bagchi. On sorting in the presence of erroneous information. *Information Processing Letters*, 43:213–215, 1992.
- 4 Elwyn R. Berlekamp. *Block coding for the binary symmetric channel with noiseless, delayless feedback*, pages 61–85. Wiley, New York, 1968.
- 5 R. Sean Borgstrom and S. Rao Kosaraju. Comparison based search in the presence of errors. In *Proceedings of the 25th annual symposium on theory of computing (STOC'93)*, pages 130–136, 1993.
- 6 Mark Braverman and Elchanan Mossel. Noisy sorting without resampling. In *Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 268–276. Society for Industrial and Applied Mathematics, 2008.
- 7 Yuval Dagan, Yuval Filmus, Ariel Gabizon, and Shay Moran. Twenty (simple) questions. In *49th ACM Symposium on Theory of Computing (STOC 2017)*, 2017.
- 8 Yuval Dagan, Yuval Filmus, Daniel Kane, and Shay Moran. The entropy of lies: playing twenty questions with a liar. *arXiv preprint*, 2018. [arXiv:1811.02177](#).
- 9 Robert Mario Fano. The transmission of information. Technical Report 65, Research Laboratory of Electronics at MIT, Cambridge (Mass.), USA, 1949.
- 10 Uriel Feige, Prabhakar Raghavan, David Peleg, and Eli Upfal. Computing with noisy information. *SIAM Journal on Computing*, 23(5):1001–1018, 1994.
- 11 Ran Gelles et al. Coding for interactive communication: A survey. *Foundations and Trends® in Theoretical Computer Science*, 13(1–2):1–157, 2017.
- 12 E. N. Gilbert and E. F. Moore. Variable-length binary encodings. *Bell System Technical Journal*, 38:933–967, 1959.
- 13 Bernhard Haeupler. Interactive channel capacity revisited. In *Foundations of Computer Science (FOCS), 2014 IEEE 55th Annual Symposium on*, pages 226–235. IEEE, 2014.
- 14 Gillat Kol and Ran Raz. Interactive channel capacity. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 715–724. ACM, 2013.
- 15 K.B. Lakshmanan, B. Ravikumar, and K. Ganesan. Coping with erroneous information while sorting. *IEEE Transactions on Computers*, 40:1081–1084, 1991.
- 16 Philip M. Long. Sorting and searching with a faulty comparison oracle. Technical Report UCSC-CRL-92-15, University of California at Santa Cruz, November 1992.
- 17 Shay Moran and Amir Yehudayoff. A note on average-case sorting. *Order*, 33(1):23–28, 2016.
- 18 Andrzej Pelc. Coding with bounded error fraction. *Ars Combinatorica*, 42:17–22, 1987.

- 19 Andrzej Pelc. Searching games with errors—fifty years of coping with liars. *Theoretical Computer Science*, 270:71–109, 2002.
- 20 Alfréd Rényi. On a problem of information theory. *MTA Mat. Kut. Int. Kozl.*, 6B:505–516, 1961.
- 21 Ronald L. Rivest, Albert R. Meyer, Daniel J. Kleitman, and Karl Winklmann. Coping with errors in binary search procedures. *Journal of Computer and System Sciences*, 20:396–404, 1980.
- 22 Leonard J Schulman. Coding for interactive communication. *IEEE transactions on information theory*, 42(6):1745–1756, 1996.
- 23 Claude Elwood Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, 1948.
- 24 Joel Spencer and Peter Winkler. Three thresholds for a liar. *Combin. Probab. Comput.*, 1(1):81–93, 1992. doi:10.1017/S0963548300000080.
- 25 Stanislaw M. Ulam. *Adventures of a mathematician*. Scribner's, New York, 1976.

# Comparing Computational Entropies Below Majority (Or: When Is the Dense Model Theorem False?)

**Russell Impagliazzo**

CSE Department, University of California San Diego, La Jolla, CA, USA  
russell@cs.ucsd.edu

**Sam McGuire**

CSE Department, University of California San Diego, La Jolla, CA, USA  
shmccguir@eng.ucsd.edu

---

## Abstract

Computational pseudorandomness studies the extent to which a random variable  $\mathbf{Z}$  looks like the uniform distribution according to a class of tests  $\mathcal{F}$ . Computational entropy generalizes computational pseudorandomness by studying the extent which a random variable looks like a *high entropy* distribution. There are different formal definitions of computational entropy with different advantages for different applications. Because of this, it is of interest to understand when these definitions are equivalent.

We consider three notions of computational entropy which are known to be equivalent when the test class  $\mathcal{F}$  is closed under taking majorities. This equivalence constitutes (essentially) the so-called *dense model theorem* of Green and Tao (and later made explicit by Tao-Zeigler, Reingold et al., and Gowers). The dense model theorem plays a key role in Green and Tao’s proof that the primes contain arbitrarily long arithmetic progressions and has since been connected to a surprisingly wide range of topics in mathematics and computer science, including cryptography, computational complexity, combinatorics and machine learning. We show that, in different situations where  $\mathcal{F}$  is *not* closed under majority, this equivalence fails. This in turn provides examples where the dense model theorem is *false*.

**2012 ACM Subject Classification** Theory of computation → Pseudorandomness and derandomization

**Keywords and phrases** Computational entropy, dense model theorem, coin problem

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.2

**Related Version** Full Version: <https://arxiv.org/abs/2011.06166>

**Funding** Research supported by NSF award 1909634 and a Simons Investigator award.

## 1 Introduction

Computational pseudorandomness is a central topic in theoretical computer science. In this scenario, one has a class  $\mathcal{F}$  of boolean functions  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  (which we’ll refer to as *tests*) and random variable  $\mathbf{Z}$  over  $\{0, 1\}^n$ . We say that  $\mathbf{Z}$  is  $\epsilon$ -pseudorandom with respect to  $\mathcal{F}$  if  $\max_{f \in \mathcal{F}} |\mathbb{E}[f(\mathbf{Z})] - \mathbb{E}[f(\mathbf{U})]| \leq \epsilon$  where  $\mathbf{U}$  is the uniform distribution over  $\{0, 1\}^n$  and  $\epsilon > 0$  is small. In this case, we think of  $\mathbf{Z}$  as “behaving like the uniform distribution” according to tests in  $\mathcal{F}$ . In general, say that two random variables  $\mathbf{X}, \mathbf{Y}$   $\epsilon$ -indistinguishable by  $\mathcal{F}$  if  $\max_{f \in \mathcal{F}} |\mathbb{E}[f(\mathbf{X})] - \mathbb{E}[f(\mathbf{Y})]| \leq \epsilon$  (and so  $\epsilon$ -pseudorandom distributions are exactly those which are  $\epsilon$ -indistinguishable from  $\mathbf{U}$ ). Constructing explicit  $\mathbf{Z}$ ’s which behave like the uniform distribution according to different test classes is among the central goals of complexity theory, with sufficiently strong constructions leading to, for example, derandomization of BPP. One way in which the theory of pseudo-randomness is rich is that there are multiple equivalent formulations of pseudo-randomness, such as Yao’s next bit test ([51]).

The various notions of pseudo-entropy and pseudo-density generalize pseudo-randomness to formalize how much randomness a distribution looks like it has as far as this class of tests can perceive. Many of these notions were first introduced as stepping stones towards



© Russell Impagliazzo and Sam McGuire;

licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 2; pp. 2:1–2:20



Leibniz International Proceedings in Informatics

LIPIcs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

pseudo-randomness, giving properties of sub-routines within constructions of pseudo-random generators. However, measuring seeming randomness quantitatively is important in many other contexts, so these notions have found wider application. For example, in mathematical subjects such as combinatorics and number theory, there is a general phenomenon of “structure vs. randomness”, where a deterministically defined object such as a graph or set of integers can be decomposed into a structured part and a random part. Pseudo-entropy quantifies how much randomness the “random part” has. Notions of pseudo-density were used in this context by Green, Tao, and Ziegler [18, 48] to show that the primes contain arbitrarily long arithmetic progressions. We can also use pseudo-entropy notions to characterize the amount of seeming randomness remains in a cryptographic key after it has been compromised with a side-channel attack. A data set used in a machine learning algorithm might not have much randomness in itself, and might not be completely random looking, but is hopefully representative of the much larger set of inputs that the results of the algorithm will be applied to, so we can use notions of pseudo-entropy to say when such algorithms will generalize. There are many possible definitions of this intuitive idea, and as with pseudo-randomness, the power of pseudo-entropy is that many of these notions have been related or proven equivalent.

In particular, the dense model theorem provides such a basic equivalence. Here, the intuitive concept we are trying to capture is the density (or relative min-entropy) of the target distribution within a larger distribution, what fraction of the larger distribution is within the target. We say that  $\mathbf{Z}$  is  $\delta$ -dense if  $\mathbb{E}[\mu(x)] = 2^{-n} \sum_x \mu(x) \geq \delta$  where  $\mu : \{0, 1\}^n \rightarrow [0, 1]$  is density function defining  $\mathbf{Z}$  (in the sense that  $\Pr[\mathbf{Z} = z] = \mu(z)/(2^n \mathbb{E}[\mu(x)])$ ). One application of indistinguishability from a dense distribution is as a stepping stone to pseudorandomness: if  $\mathbf{Z}$  is indistinguishable from a distribution  $\mathbf{M}$  with density  $\delta$  within the uniform distribution, then applying a randomness extractor with min-entropy rate  $n - \log(1/\delta)$  to  $\mathbf{Z}$  is a pseudorandom distribution. A more sophisticated application comes from additive number theory. It is not hard to show that a *random* subset of  $[N] = \{1, 2, \dots, N\}$  (including each element with probability  $1/2$ , say) contains many arithmetic progressions (which are sets of the form  $\{a, a + b, a + 2b, a + 3b, \dots\}$ ). Szemerédi [45] showed that, in fact, sufficiently *dense* subsets of the integers also contain such arithmetic progressions: specifically, that for any  $k$ , the size of the largest subsets of  $[N]$  which *doesn't* contain an arithmetic progression grows like  $o(N)$ .

So we would like some technology to reason about random variables  $\mathbf{Z}$  which “behave like dense distributions”. It turns out, however, that formalizing what it means for  $\mathbf{Z}$  to “behave like a dense distribution” is subtle. Here are three perfectly legitimate candidates:

**Candidate 1:**  $\mathbf{Z}$  behaves like a  $\delta$ -dense distribution if it behaves like something that's  $\delta$ -dense.

Formally, this means that  $\mathbf{Z}$  is  $\varepsilon$ -indistinguishable from some  $\delta$ -dense distribution. In this case, we say that  $\mathbf{Z}$  has a  $\delta$ -dense  $\varepsilon$ -model.

**Candidate 2:**  $\mathbf{Z}$  behaves  $\delta$ -dense if it's  $\delta$ -dense inside of something that behaves like the uniform distribution. Formally this means there's an  $\varepsilon$ -pseudorandom distribution  $\mathbf{X}$  in which  $\mathbf{Z}$  is  $\delta$ -dense. In this case, we say that  $\mathbf{Z}$  is  $\delta$ -dense in an  $\varepsilon$ -pseudorandom set.

**Candidate 3:**  $\mathbf{Z}$  behaves  $\delta$ -dense if it appears to be the case that conditioning on  $\mathbf{Z}$  increases the size of any set by at most (roughly) a  $1/\delta$ -factor. This is an operational definition: conditioning on a (truly) dense set increases the set by at most a  $1/\delta$ -fraction, so we should expect the same behavior from things that behave like a dense set. Formally, this means that  $\delta \mathbb{E}[f(\mathbf{Z})] \leq \mathbb{E}[f(\mathbf{U})] + \varepsilon$  for any  $f$  in our test class  $\mathcal{F}$ . In this case, we say that  $\mathbf{Z}$  has  $(\varepsilon, \delta)$ -pseudodensity.

Precisely which definition you pick will depend on what you know about  $\mathbf{Z}$  and in what sense you would like it to behave like a  $\delta$ -dense distribution. Indeed, each of these definitions have appeared in different applications ([25], [18], [13], respectively), so there are scenarios where each of these types of behavior is desired. In general, the first candidate is the strongest (and, arguably, the most natural), but it is sometimes hard to establish that a distribution has the property. The following claim gives some simple relationships between the definitions:

▷ **Claim 1.** For any  $\mathcal{F}$ , the following hold:

1. If  $\mathbf{Z}$  has a  $\delta$ -dense  $\varepsilon$ -model, then  $\mathbf{Z}$  is  $\delta$ -dense in a  $\varepsilon$ -pseudorandom set.
2. If  $\mathbf{Z}$  is  $\delta$ -dense in an  $\varepsilon$ -pseudorandom set, then  $\mathbf{Z}$  has  $(\varepsilon, \delta)$ -pseudodensity.

Proof sketch.

1. Let  $\mathbf{M}$  be the  $\delta$ -dense  $\varepsilon$ -model for  $\mathbf{Z}$ . Note that  $\mathbf{U} = \delta\mathbf{M} + (1-\delta)\overline{\mathbf{M}}$ . So  $\mathbf{U}' = \delta\mathbf{Z} + (1-\delta)\overline{\mathbf{M}}$  is  $\varepsilon$ -pseudorandom and  $\mathbf{Z}$  is  $\delta$ -dense within it.
2. Suppose  $\mathbf{Z}$  is  $\delta$ -dense in  $\mathbf{Z}'$  which  $\varepsilon$ -pseudorandom for  $\mathcal{F}$ . Then for any  $f \in \mathcal{F}$ ,  $\delta\mathbb{E}[f(\mathbf{Z})] \leq \mathbb{E}[f(\mathbf{Z}')] \leq \mathbb{E}[f(\mathbf{U})] + \varepsilon$ .  $\triangleleft$

The marvelous quality of these three candidates in particular is that, for many natural  $\mathcal{F}$ , all of them are *equivalent*, and so establishing even  $(\varepsilon', \delta)$ -pseudodensity is enough to guarantee the existence of a  $\delta$ -dense  $\varepsilon$ -model.

This equivalence holds for  $\mathcal{F}$  which are *closed under majority*, meaning for any  $k$  (which we can think of as  $k = O(1)$  for now), if  $f_1, \dots, f_k \in \mathcal{F}$  then  $\text{MAJ}_k(f_1, \dots, f_k) \in \mathcal{F}$ , where  $\text{MAJ} : \{0, 1\}^n \rightarrow \{0, 1\}$  is 1 if at least half of its input bits are 1. In fact, it holds for more general  $\mathcal{F}$  if we allow the distinguishing parameter ( $\varepsilon'$  in  $(\varepsilon', \delta)$ -pseudodensity) to be exponentially small (as in the original formulation, which we'll discuss later on). In this case, the subtlety in defining what it means to behave like a dense set vanishes. These equivalences constitute (essentially) what is known as the *dense model theorem*, originating in the work of Green-Tao [18] and Tao-Zeigler [48], and independently in Barak et al. [8] (though in different guises). This result has been fruitfully applied in many seemingly unrelated areas of mathematics and computer science: additive number theory [18, 48] where  $\mathcal{F}$  encodes additive information about subsets of  $\{1, \dots, N\}$  (or possibly a more general group), graph theory [49, 38] where  $\mathcal{F}$  encodes cuts in a fixed graph, circuit complexity [49], Fourier analysis [29], machine learning [29] and leakage-resilient cryptography [14]. The ubiquity of the dense model theorem motivates a simple question: are there natural scenarios in which the dense model theorem is *false*?

We show that the answer to this question is *yes*. In particular, we show that for either implication from Claim 1 there is a class  $\mathcal{F}$  and a random variable  $\mathbf{Z}$  so that converse fails to hold. From the computational entropy perspective, we show that the three computational entropies we've discussed are inequivalent for certain test classes  $\mathcal{F}$ . Necessarily (with  $\varepsilon'$  not exponentially small) these classes are *not* closed under majority and so we will need to look “below” majority in order to find our counterexamples.

## 1.1 The dense model theorem

We turn to discuss the dense model theorem in some more detail to better contextualize our work. Restricting our attention to random variable over  $\{0, 1\}^n$ , the dense model theorem states the following:

► **Theorem 1.1** (Dense model theorem). *Let  $\mathcal{F}$  be a class of tests  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  and  $\mathbf{Z}$  a random variable over  $\{0, 1\}^n$  with  $(\varepsilon\delta, \delta)$ -pseudodensity with respect to  $\text{MAJ}_k \circ \mathcal{F}$  for  $k = O(\log(1/\delta)/\varepsilon^2)$ . Then  $\mathbf{Z}$  has a  $\delta$ -dense  $\varepsilon$ -model with respect to  $\mathcal{F}$ .*



We will generally also consider a parameter  $\varepsilon'$ , which in this case is  $\varepsilon\delta$ , the additive error in pseudodensity. To get an intuition for what this is saying, let's consider a setting where it's false but for trivial reasons. As a simple example given in [52], pick a set  $\mathbf{Z}$  some set as a  $(1 - \varepsilon)$  fraction of another set  $\mathbf{S}$  of size  $\delta 2^n$ . Then  $\mathbf{Z}$  doesn't have a  $\delta$ -dense  $\varepsilon$ -model (i.e.  $\mathbf{S}$ ) with respect to  $\mathbf{Z}$ 's indicator function, which we'll call  $f$ . On the other hand, the distribution  $\mathbf{W}$  obtained by sampling  $\mathbf{Z}$  with probability  $\delta$  and sampling from  $\mathbf{S}$ 's complement with probability  $1 - \delta$  is at most  $\varepsilon\delta$ -distinguishable from  $\mathbf{S}$  for any function, since  $\varepsilon\delta$  is simply the measure of the difference between  $\mathbf{S}$  and  $\mathbf{Z}$ . In particular  $\mathbf{Z}$  is  $\delta$ -dense in the  $\varepsilon\delta$ -pseudorandom  $\mathbf{W}$  (which implies, via Claim 1, that it is  $(\varepsilon\delta, \delta)$ -pseudodense). This means that the Theorem 1.1 is tight for the dependence on  $\varepsilon' = \varepsilon\delta$ , in that it becomes false for  $\Omega(\varepsilon\delta)$ . In many instances, we think of  $\varepsilon = 1/\text{poly}(n)$ ,  $\delta$  constant (or perhaps with mild dependences on  $n$ ) and  $\varepsilon' = \delta\varepsilon$ .

Originally, the dense model theorem was proved with a different (and stronger) assumption; namely, that  $\mathbf{Z}$  is dense in a pseudorandom set. Green and Tao, in proving that the primes contain arbitrarily long arithmetic progressions, used it to the following effect: if  $\mathbf{Z}$  are the prime numbers up to  $n$ , then its density is known to behave like  $\Theta(1/\log n)$ . On the other hand, Szemerédi [45] showed that sufficiently dense subsets of  $\mathbb{Z}$  contain arbitrarily long arithmetic progressions. The best bounds for Szemerédi's theorem require density  $\omega(1/\log \log n)$ , which is much larger than the primes (see [16] and the recent [9] for more on the rich history on this and related problems). Not all is lost, however: the only property of dense sets that we're interested in is that they contain arithmetic progressions. So Green and Tao construct a class  $\mathcal{F}$  of tests which can “detect” arithmetic progressions and under which the primes are dense inside of a  $\mathcal{F}'$ -pseudorandom set (more on  $\mathcal{F}'$  later). By applying the dense model theorem, we conclude that the primes “look like” a dense set (themselves having long arithmetic progressions) with respect to the class  $\mathcal{F}$ . As  $\mathcal{F}$  detects arithmetic progressions, it must be the case that the primes possess them. Of course, many details need to be filled in, but we hope this example shows the reader the “spirit” of the dense model theorem.

A primary source of interest in the dense model theorem is in the connections it shares with seemingly unrelated branches of mathematics and computer science. The original application was in additive number theory, but it was independently discovered and proved in the context of cryptography ([8, 14]). RTV [38] and Gowers [17] observed proofs of the dense model theorem which use linear programming duality, which is in turn related to Nisan's proof of the hardcore lemma from circuit complexity [28]. In fact, Impagliazzo [29] shows in unpublished work that optimal-density versions of the hardcore lemma due to Holenstein [26] actually *imply* the dense model theorem. Klivans and Servedio [32] famously observed the relationship between the hardcore lemma and *boosting*, a fundamental technique for aggregating weak learners in machine learning [15]. Together with the result of Impagliazzo, this connection means that dense model theorems can be proved by a particular type of boosting algorithm. A boosting argument for the existence of dense models also gives us *constructive* versions of the dense model theorem, which are needed for algorithmic applications. Zhang [52] (without using Impagliazzo's reduction from the dense model theorem to the hardcore lemma) used the boosting algorithm of [7] directly to prove the dense model theorem with optimal query complexity ( $k$ ).

In addition to its connections to complexity, machine learning, additive number theory and cryptography, the dense model theorem (and ideas which developed from the dense model theorem, chiefly the approximation theorem of [49]), have been used to understand the weak graph regularity lemma of Frieze and Kannan [29], notions of computational differential



privacy [36] and even generalization in generative adversarial networks (GANs) [5]. We now turn to discussing the complexity-theoretic aspects of the dense model theorem, specifically regarding our question of whether the  $\text{MAJ}_k$  from the statement is optimal.

As alluded to earlier, Green and Tao actually worked in a setting where  $\mathcal{F}'$  doesn't need to compute majorities but where  $\varepsilon\delta$  (that is, the distinguishing parameter in the pseudodensity assumption in the statement of Theorem 1.1) needs to be replaced by some  $\varepsilon' = \exp(-\text{poly}(1/\varepsilon, 1/\delta))$  (with  $k = \text{poly}(1/\delta, 1/\varepsilon)$  experiencing a small increase). We state this result, as proved in Tao and Zeigler [48] and stated this way in RTTV [38], for comparison. For a test class  $\mathcal{F}$ , let  $\prod_k \mathcal{F}$  be the set of tests of the form  $\prod_{i \in [k]} f_i$  for  $f_i \in \mathcal{F}$ .

► **Theorem 1.2** (Computationally simple dense-model theorem, strong assumption). *Let  $\mathcal{F}$  be a class of tests  $f : \{0, 1\}^n \rightarrow [0, 1]$  and  $\mathbf{Z}$  a random variable over  $\{0, 1\}^n$  which is  $\delta$ -dense in a set  $\varepsilon'$ -pseudorandom for  $\prod_k \mathcal{F}$  with  $k = \text{poly}(1/\delta, 1/\varepsilon)$  and  $\varepsilon' = \exp(-1/\delta, 1/\varepsilon)$ . Then  $\mathbf{Z}$  has a  $\delta$ -dense  $\varepsilon$ -model with respect to  $\mathcal{F}$ .*

RTTV [38] observe that this proof can be adapted to work for  $\varepsilon'$  have polynomial dependence on  $\varepsilon, \delta$  by restricting to the case of boolean-valued tests. Doing so, however, makes  $\mathcal{F}'$  much more complicated (essentially requiring circuits of size exponential in  $k$ ). In Theorem 1.1, we can obtain the best of both worlds:  $\varepsilon'$  has polynomial dependence on  $\varepsilon, \delta$  and the complexity blow-up is rather small. However, in this more picturesque circumstance, we need to be able to compute majorities. Is such a tradeoff necessary? Our results suggest that the answer is yes. Theorem 1.6 (stated in the following section) tells us that if the dense model theorem is true for  $\mathcal{F}$ , then there's a small, constant-depth circuit with  $\mathcal{F}$ -oracle gates approximating majority on  $O(1/\varepsilon^2)$  bits.

Another important aspect of the dense model theorem is how the different assumptions are related. As mentioned, the original assumption was that  $\mathbf{Z}$  is  $\delta$ -dense in an  $\varepsilon$ -pseudorandom set, but the proof can be extended to the case where  $\mathbf{Z}$  is  $(\varepsilon, \delta)$ -pseudodense. Claim 1 showed that the former assumption implies that latter assumption. When the dense model theorem is true, the latter also implies the former: simply apply the dense model theorem to  $\mathbf{Z}$  which is  $(\varepsilon, \delta)$ -dense to obtain a  $\delta$ -dense  $\varepsilon$ -model. Then, by the first part of Claim 1, we're done.

First, we give examples of situations where these two notions are distinct. For example, we show in Theorem 1.4 and Theorem 1.5 that they are inequivalent when  $\mathcal{F}$  is constant-depth polynomial size circuits or when  $\mathcal{F}$  is a low-degree polynomial over a finite field. Note that a separation between pseudodensity and being dense in a pseudorandom set also implies a separation between pseudodensity and having a dense model, as being dense in a pseudorandom set is a necessary condition for having a dense model.

Second, we show that the dense model theorem is false even when we make the stronger assumption that the starting distribution  $\mathbf{Z}$  is dense in a pseudorandom set. Specifically, in Theorem 1.3 we can show that some distributions  $\mathbf{Z}$  are dense in a pseudorandom set but fail to have a dense model when  $\mathcal{F}$  consists of constant-depth, polynomial size circuits.

Having contextualized our work some, we now turn to describe our contributions in more detail.

## 1.2 Contributions

We separate the previously described notions of computational entropy, giving examples where the dense model theorem is false. We are able to prove different separations when  $\mathcal{F}$  is constant-depth unbounded fan-in circuits, low-degree polynomials over a finite field, and, in one case, any test class  $\mathcal{F}$  which cannot efficiently approximate majority (in some sense made explicit later on). The only known separation prior was between pseudodensity and having a dense model for bounded-width read-once branching programs, due to Barak et al. [8].

Let  $\mathcal{C}(S, d)$  denote the class of unbounded fan-in, size  $S$ , depth  $d$  circuits. We are generally thinking of  $S = \text{poly}(n)$  and  $d = O(1)$ , which corresponds to the complexity class  $\text{AC}^0$ . Theorem 1.3 shows that  $\mathbf{Z}$  being  $\delta$ -dense in an  $\varepsilon$ -pseudorandom set need not imply that  $\mathbf{Z}$  has a  $\delta$ -dense  $\varepsilon$ -model when the test class is  $\mathcal{C}(S, d)$ :

► **Theorem 1.3.** *Let  $\varepsilon, \varepsilon' > 0$  be arbitrary,  $\delta \geq \varepsilon'/8$  and*

$$S \leq \exp\left(O\left(\frac{\sqrt{\varepsilon'}}{\varepsilon} \cdot \frac{\sqrt{\log(1/\delta)}}{\log(1/\varepsilon')}\right)^{1/(d-1)}\right).$$

*Then for  $\mathcal{F} = \mathcal{C}(S, d)$ , there is a random variable  $\mathbf{D}$  over  $\{0, 1\}^n$  with  $n = O(\log(1/\delta)/\varepsilon^2)$  so that  $\mathbf{D}$  is  $\delta$ -dense in an  $\varepsilon'$ -pseudorandom set but does not have a  $\delta$ -dense  $\varepsilon$ -model. In particular, the dense model theorem is false in this setting.*

Recall that the dense model theorem is false when  $\varepsilon' = \Omega(\varepsilon\delta)$ , which makes the restriction  $\delta \geq \varepsilon'/8$  extremely mild. A common regime is  $\varepsilon = 1/\text{poly}(n)$ ,  $\delta = O(1)$  and  $\varepsilon' = \delta\varepsilon = \Theta(\varepsilon)$ , in which case this gives us (essentially) a lower bound of weakly exponential in  $1/\sqrt{\varepsilon} \approx 1/\sqrt{\varepsilon'}$ .

Let  $\mathbf{N}_\alpha$  denote the product distribution of  $n$  Bernoulli random variables with success probability  $1/2 - \alpha$ . Recall that density in a pseudorandom set readily implies pseudodensity, and one can use the dense model theorem to show the converse. We show that  $(\varepsilon, \delta)$ -pseudodensity need not imply  $\delta$ -density in an  $\varepsilon$ -pseudorandom set when the test class is  $\mathcal{C}(S, d)$ :

► **Theorem 1.4.** *Fix  $\varepsilon, \varepsilon', \delta > 0$ ,  $d \in \mathbb{N}$ , and*

$$S \leq \exp\left(O\left(\frac{\sqrt{\delta}}{\sqrt{\varepsilon}} \cdot \frac{\log(1/\delta)}{\log(1/\varepsilon')}\right)^{1/(d-1)}\right).$$

*Then  $\mathbf{N}_{\sqrt{\varepsilon/\delta}}$  over  $\{0, 1\}^n$  with  $n = O(1/\varepsilon)$  is  $(\varepsilon', \delta)$ -pseudodense and yet  $\mathbf{N}_{\sqrt{\varepsilon/\delta}}$  is not  $\delta$ -dense inside of any  $\varepsilon$ -pseudorandom set.*

The dependence  $\varepsilon'$  means that we can take  $\varepsilon'$  exponentially smaller than  $\varepsilon$  and still obtain a separation. This case corresponds to  $\mathcal{F}$  being “very” fooled by  $\mathbf{N}_\alpha$  but still not being  $\delta$ -dense in a “mildly” pseudorandom set. This result draws on a recent line of work in the pseudorandomness literature – often referred to as “the coin problem” and studied in, e.g., [42, 12, 1, 46] – which concerns the ability of a test class  $\mathcal{F}$  unable to compute majority has in distinguishing  $\mathbf{N}_\alpha$  and  $\mathbf{U}$ . We will discuss this connection in more detail during the proof overviews.

We prove a similar separation for degree- $d$   $\mathbb{F}_p$ -polynomials (on  $n$  variables), which generalizes (and uses techniques from) a recent result of Srinivasan [44] in the case where  $\delta = 1$ . In this case, we think of a distribution  $\mathbf{Z}$  as being  $(\varepsilon', \delta)$ -pseudodense for degree- $d$   $\mathbb{F}_p$ -polynomials when  $\delta \Pr[P(\mathbf{Z}) \neq 0] - \varepsilon' \geq \Pr[P(\mathbf{U}) \neq 0]$  for any degree- $d$  polynomial  $P \in \mathbb{F}_p[X_1, \dots, X_n]$  (noting that we are only evaluating  $P$  over  $\{0, 1\}^n$ ).

► **Theorem 1.5.** *Fix a finite field  $\mathbb{F}$  with characteristic  $p = O(1)$ ,  $\varepsilon, \varepsilon' > 0$  and let  $c > \delta > 0$  where  $c \approx 1/200$  is an absolute constant. Suppose that*

$$d \leq O(\sqrt{\delta/\varepsilon}).$$

*Then when  $\mathcal{F}$  is the  $n$ -variate degree- $d$  polynomials over  $\mathbb{F}$  with  $n = 1/\varepsilon$ , and  $\alpha = O(\sqrt{\varepsilon/\delta})$ ,  $\mathbf{N}_\alpha$  is  $(\varepsilon', \delta)$ -pseudodense but is not  $\delta$ -dense inside of an  $\varepsilon$ -pseudorandom set.*

This implies lower bounds for constant-depth circuits with  $\text{MOD}_p$  gates by the classical lower bounds of Razborov [37] and Smolensky [43]. Perhaps more interestingly, this holds even over non-prime fields. Also notably, there is no dependence on  $\varepsilon' \leq \varepsilon\delta$ , so we can take it to be arbitrarily small.

We also prove a more general separation between pseudodensity and density in a pseudorandom set. This result, drawing from the work of [42], provides a more specific characterization of the sense in which dense model theorems are “required” to compute majority.

► **Theorem 1.6.** *Let  $\varepsilon, \delta > 0$ . Suppose  $\mathcal{F}$  is a test class of boolean functions  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  with the following property: there is no  $\text{AC}^0$   $\mathcal{F}$ -oracle circuit of size  $\text{poly}(n \cdot \frac{\sqrt{\delta}}{\varepsilon^{3/2}})$  computing majority on  $O(\sqrt{\delta/\varepsilon})$  bits.*

*Then  $\mathbf{N}_{\sqrt{\varepsilon/\delta}}$  is  $(\varepsilon\delta, \delta)$ -pseudodense and yet does not have a  $\delta$ -dense  $\varepsilon$ -model. In particular, when the hypotheses are met, the dense model theorem is false.*

Informally, this says that any  $\mathcal{F}$  which can refute the pseudodensity of  $\mathbf{N}_\alpha$  is only “a constant-depth circuit away” from computing majority.

### 1.3 Related work

#### Computational entropy

Computational entropy was studied systematically in [8] and is relevant to various problems in complexity and cryptography such as leakage-resilience [14], constructions of PRGs from one-way functions [25, 21, 20]. and derandomization [13].

There are a number of definitions of computational entropy which we *don't* consider in this work. For example, Yao pseudoentropy [51] (see also [8]), corresponding to random variables which are “compressible” by a class of tests  $\mathcal{F}$ , in the sense that  $\mathcal{F}$  can encode and decode the random variable by encoding into a small number of bits. Yao pseudoentropy was recently used in time-efficient hardness-to-randomness tradeoffs [13], where (randomness-efficient) samplers for pseudodense distributions were used with an appropriate extractor to construct a pseudorandom distribution. Another example is *inaccessible entropy* of Haitner et al. [21], corresponding to the entropy of a message at some round in a two-player protocol conditioned on the prior messages and the randomness of the players, which is used in efficient constructions of statistically hiding commitment schemes from one-way functions [20].

Separating notions of computational entropy has been studied before in [8], who prove a separation of pseudodensity and having a dense model for bounded-width read-once branching programs. Separating notions of *conditional* computational entropy was studied in [27], showing separations between conditional variants of Yao pseudoentropy and having a dense model.

As mentioned in [27], citing [49] and personal communication with Impagliazzo, another question of interest is whether Yao pseudoentropy (corresponding to efficient encoding/-decoding algorithms) implies having dense model. It is not hard to see that small Yao pseudoentropy implies small pseudodensity, with some mild restrictions on  $\mathcal{F}$ . It would be interesting to see if the techniques from this paper can be used to understand Yao pseudoentropy in more detail. We leave this to future work.

#### Complexity of dense model theorems and hardness amplification

Prior work on the complexity of dense model theorems has included a tight lower bound on the query complexity [52] and a lower bound on the advice complexity [50]. As far as we are aware, this is the first work to consider the computational complexity of dense model theorems.

There has also been prior work on the computational complexity of hardness amplification, establishing that various known strategies for hardness amplification require the computation of majority [34, 42, 19, 41]. It is known that a particular type of hardness amplification given by the *hardcore lemma* implies the dense model theorem [29].

Our results are stronger in the following sense: previous work [34, 42, 19] shows that *black-box hardness amplification proofs* require majority. This means that if you amplify the hardness of  $f$  in some black-box way, then this can be used to compute majority. In our case, we simply show (in different settings) that the dense model theorem is *false*, regardless of how we tried to prove it. By the connection between the hardcore lemma and the dense model theorem, our results also provide scenarios where the hardcore lemma is false. As far as we are aware, these are the first such scenarios recorded in the literature.

## 1.4 Technical overview

We discuss two general themes that appear consistently in the proofs and then discuss each of the main theorems in some more detail.

### 1.4.1 Dense distributions have mostly unbiased bits

A commonly-used observation in theoretical computer science is that most bit positions of a  $\delta$ -dense random variable over  $\{0, 1\}^n$  have bias  $O(\sqrt{\log(1/\delta)}/n)$  (see, for example, the introduction of [35]). Relevant to our purposes, it provides a *necessary* condition for having a  $\delta$ -dense  $\varepsilon$ -model with respect to any class  $\mathcal{F}$  containing the projections  $z \mapsto z_i$ .  $\mathbf{Z}$  has a  $\delta$ -dense  $\varepsilon$ -model, then most bits of  $\mathbf{Z}$  have bias  $\varepsilon + O(\sqrt{\log(1/\delta)}/n)$ . In particular, if all of the bits of  $\mathbf{Z}$  have *large* bias, then it can't have a dense model.

This is used directly in the proof of Theorem 1.3. In this case, we construct a distribution  $\mathbf{Z}$  which is  $\delta$ -dense in a set which is  $\varepsilon$ -pseudorandom for  $\text{AC}^0$  but where the each bit is noticeably biased away from  $1/2$ .

In order to prove separations between pseudodensity and being dense in a pseudorandom set – as in Theorem 1.4, Theorem 1.5 and Theorem 1.6 – we need to consider the bias of larger subsets of variables. Considering just two bits is sufficient to prove mild concentration bounds on the weight of pseudorandom strings. This implies that the tails of dense subsets of pseudorandom sets should not be too heavy.

### 1.4.2 Biased coin distribution

The *biased coin distribution*,  $\mathbf{N}_\alpha$  over  $\{0, 1\}^n$  is the product of  $n$  Bernoulli random variables with success probability  $1/2 - \alpha$ .  $\mathbf{N}_\alpha$  has recently garnered significant interest in the pseudorandomness literature (see [2, 12, 46, 10, 1]). Shaltiel and Viola [42] showed that if  $f$  is a test which  $\varepsilon$ -distinguishes  $\mathbf{N}_\alpha$  from  $\mathbf{U}$ , then there is a small, constant-depth circuit  $C$  with  $f$ -oracle gates which computes majority on  $O(1/\varepsilon)$  bits. A similar, but qualitatively different, connection due to Limaye et al [33] – extended to any choice of  $\varepsilon > 0$  by Srinivasan [44] – shows that any  $\mathbb{F}_p$ -polynomial with advantage  $1 - 2\varepsilon$  in distinguishing  $\mathbf{N}_\alpha$  from  $\mathbf{U}$  must have degree  $\Omega(\log(1/\varepsilon)/\alpha)$ . We extend some of these pseudorandomness results regarding  $\mathbf{N}_\alpha$  to *pseudodensity* results.

First, we extend the observation of Shaltiel and Viola to apply to tests  $f$  for which  $\mathbb{E}[f(\mathbf{Z})] \geq \delta \mathbb{E}[f(\mathbf{U})] + \varepsilon$  (which corresponds to pseudorandomness when  $\delta = 1$ ). This gives us unconditional pseudodensity for test classes  $\mathcal{F}$  which can't be used in small, constant-depth oracle circuits approximating majority. We also extend the observation of [33] to show lower bounds on the  $\mathbb{F}_p$ -degree for any function  $f$  which refutes the pseudodensity of  $\mathbf{N}_\alpha$ .

In Lemma 13, we show that  $\mathbf{N}_\alpha$  exhibits  $(\varepsilon, \delta)$ -pseudodensity for  $\varepsilon = (p \cdot O(\log S)^{d-1})^k$  and  $\delta = e^{-\alpha k/p}$ . This can be seen as a generalization of Tal's result, building on [12, 1, 42] that  $\mathbf{N}_\alpha$  is  $3\alpha \cdot O(\log S)^{d-1}$ -pseudorandom for  $\mathcal{C}(S, d)$ .

Tal uses a Fourier analytic proof which becomes very simple given tail bounds on the Fourier spectrum of  $\mathbf{AC}^0$  (the latter being the main contribution of [46]). More generally, any  $\mathcal{F}$  enjoying sufficiently strong tail bounds on the Fourier spectrum (in the  $\ell_1$  norm) cannot distinguish between  $\mathbf{N}_\alpha$  and uniform. It turns out, as proved by Tal and recorded in Agarwal [2], that if  $\mathcal{F}$  is closed under restrictions then even bounding the first level of the Fourier spectrum works. The proof of Lemma 13 based specifically on the switching lemma for constant-depth circuits. While switching lemmas can be used to show Fourier concentration, it would be interesting to find a proof which only uses the assumption of Fourier concentration (or some Fourier-analytic assumption).

### 1.4.3 Theorem 1.3

Our goal is to construct a random variable  $\mathbf{D}$  which is dense inside of an  $\mathbf{AC}^0$ -pseudorandom set but where each bit is biased away from 0. In this case,  $\mathbf{D}$  would be distinguishable from any dense set, since the average bit of a dense set is roughly unbiased. Doing so requires two steps.

The first step is constructing an appropriate distribution  $\mathbf{Z}$  that fools  $\mathbf{AC}^0$  circuits. For this we adopt a general strategy of Ajtai and Wigderson [3] (and applied in many contexts in pseudorandomness since; see, e.g., [40]): to fool a circuit  $C$ , we start by producing a random restriction to simplify  $C$  to a short decision tree (via the switching lemma), and then we fool the decision tree on the remaining bits using a  $k$ -wise independent distribution  $\mathbf{S}$ . If we wanted  $\mathbf{Z}$  to have small support size, we would need some way of producing random restrictions with a small amount randomness (which is precisely the approach of Ajtai-Wigderson and later work). Fortunately, we only care about the existence of  $\mathbf{Z}$  and are therefore content to use the “non-derandomized” switching lemma.

The second step is finding a dense subset  $\mathbf{D}$  of  $\mathbf{S}$  with biased bits. We do this by constructing  $\mathbf{S}$  so that each bit has bias roughly  $\sqrt{\log(1/\delta)/K}$ , where  $k \ll K \ll n$  is a parameter. This is achieved by randomly bucketing the indices into  $K$  buckets and assigning each bucket a random bit, which reduces the dimension of the problem from  $n$  to  $K$ . This means we can pick a  $\delta$ -dense event in  $\{0, 1\}^K$  with extremal bias – met (up to constants) by the function accepting all strings with weight less than  $K/2 - K\sqrt{\log(1/\delta)}$  – in order to find a dense subset of  $\mathbf{S}$  with large bias. The bucketing construction introduces some error when a small set  $I \subseteq [n]$  hits to distinct elements in some buckets.

### 1.4.4 Theorem 1.4

We will show  $\mathbf{N}_\alpha$  has  $(\delta, \varepsilon')$ -pseudodensity for  $\mathbf{AC}^0$  for  $\delta = \varepsilon' = O(1)$ ,  $\alpha = 1/\text{poly} \log(n)$ . The idea is that  $\mathbf{N}_\alpha$  can be sampled by first sampling a random restriction which leaves a  $p$  fraction of the bits unset (and is unbiased on the restricted bits) and then setting the remaining bits with bias  $\alpha/p$ . Applying the switching lemma, we conclude that  $\mathbb{E}[f(\mathbf{N}_\alpha)] \approx \mathbb{E}[f'(\mathbf{N}_{\alpha/p})]$  where  $f'$  is a short decision tree (which doesn't not depend on all of its inputs). A simple calculation reveals that acceptance probability of  $f'$  can increase by at a most a factor  $(1 + \alpha/p)^d \leq e^{\alpha d/p}$  when passing from the uniform distribution to  $\mathbf{N}_{\alpha/p}$ . By incorporating the error from the switching lemma (i.e. the advantage lost by conditioning on the switching lemma succeeding), we get  $(\delta, \epsilon)$ -pseudodensity.

To prove the separation, we use the fact that the Hamming weight of a random variable fooling  $\mathcal{C}(S, d)$  is concentrated around its expectation. This means in particular that if  $\mathbf{N}_\alpha$  were  $\delta$ -dense in a pseudorandom distribution, then the tails of  $\mathbf{N}_\alpha$  couldn't be too heavy and therefore  $\alpha$  couldn't be too large.

### 1.4.5 Theorem 1.5 and Theorem 1.6

Theorem 1.5 and Theorem 1.6 draw from related work of Srinivasan [44] and Shaltiel-Viola [42] respectively.

With  $\epsilon > 0$  and  $\mathcal{F}$  an arbitrary class of tests  $f : \{0, 1\}^n \rightarrow \{\pm 1\}$ , suppose that  $f \in \mathcal{F}$  witnesses that  $\mathbf{N}_\epsilon$  fails to have  $(\epsilon', \delta)$ -pseudo-density in the sense that

$$\mathbb{E}[f(\mathbf{U})] \leq \delta \mathbb{E}[f(\mathbf{N}_\beta)] - \gamma.$$

[44] and [42] both make use of the following simple observation. Given two strings  $u, v \in \{0, 1\}^m$  with  $\text{wt}(u) = (1/2 - \epsilon)m$  and  $\text{wt}(v) = m/2$ , a uniformly random index  $i \in [m]$  has  $u_i$  distributed as a  $(1/2 - \epsilon)$ -biased coin and  $v_i$  as an unbiased coin. In our case, applying  $f$  to sufficiently many random samples from  $u$  or  $v$  “distinguishes” the two of them, but in a weaker sense.

In the case of Theorem 1.6, we can amplify acceptance probabilities by increasing the size of the circuit by a factor  $1/\epsilon\delta$ , after which we can apply [42] saying that constant-error distinguishers between  $\mathbf{N}_\alpha$  and  $\mathbf{U}$  can be used to compute majority.

For Theorem 1.5, we apply a beautiful recent result of Srinivasan [44] showing that any  $m$ -variate polynomial (over a finite field) which vanishes on most points on the slice  $1/2 - \alpha$  and doesn't vanish on most points on the slice  $1/2$  must have high degree  $\Omega(\alpha m)$ . One way of interpreting this result is that low-degree polynomials can't approximately solve certain “promise” versions of majority.

In this latter case, we need to open up the error reduction procedure we use for Theorem 1.6 and show how to approximate it using low-degree polynomials. This will ultimately be achieved by approximating OR with a probabilistic polynomial, as in [37, 43]. The detailed proofs of these results are deferred to the full version of the paper.

## 2 Technical tools

We write  $[n] = \{1, \dots, n\}$  and use boldface to denote random variables. Let  $\mathcal{C}(S, d)$  be the set of size  $S$ , depth- $d$  unbounded fan-in circuits. For a boolean function  $f : \{0, 1\}^n \rightarrow \{0, 1\}$ , let  $DT(f)$  denote the depth of the shortest decision tree computing  $f$ .

### 2.1 Biased coins

As before, let  $\mathbf{N}_\alpha$  denote the random variable corresponding to the product of  $n$  independent coins with bias  $(1/2 - \alpha)$ . That is,

$$\Pr[\mathbf{N}_\alpha = z] = (1/2 - \alpha)^{\text{wt}(z)} (1/2 + \alpha)^{n - \text{wt}(z)}$$

where  $\text{wt}(z)$  denotes the Hamming weight of  $z$ .

For a random variable  $\mathbf{Z}$  over  $\{0, 1\}^n$  and  $i \in [n]$ , let  $\text{bias}_i(\mathbf{Z}) = |\Pr[\mathbf{Z}_i = 1] - \Pr[\mathbf{Z}_i = 0]|/2$ . Let  $\mathcal{B} = \{z \mapsto z_i : i \in [n]\}$  be the set of monotone projections. A random variable  $\mathbf{Z} = (\mathbf{Z}_1, \dots, \mathbf{Z}_n)$  is  $\epsilon$ -pseudorandom with respect to  $\mathcal{B}$  precisely when each marginal  $\mathbf{Z}_i$  has the property that  $\text{bias}_i(\mathbf{Z}) = |\Pr[\mathbf{Z}_i = 1] - 1/2| \leq \epsilon$  for each  $i \in [n]$ . In particular,

▷ **Claim 2.** For any  $\epsilon > 0$ ,  $\mathbf{N}_\epsilon$  is  $\epsilon$ -pseudorandom with respect to  $\mathcal{B}$ .

## 2.2 Information theory

The (*Shannon*) *entropy* of a random variable is defined as

$$H(\mathbf{Z}) = - \sum_{x \in \{0,1\}^n} p_{\mathbf{Z}}(x) \log p_{\mathbf{Z}}(x),$$

where  $p_{\mathbf{Z}}$  is the probability density function corresponding to  $\mathbf{Z}$ . The Shannon entropy of random vector is sub-additive, in that  $H(\mathbf{Z}) \leq \sum_{i \in [n]} H(\mathbf{Z}_i)$ . When  $\mathbf{Z} \in \{0,1\}$  and  $\Pr[\mathbf{Z} = 1] = p$ , we use  $h(p) = H(\mathbf{Z}) = -(p \log p + (1-p) \log(1-p))$  to denote the binary entropy function.

The *min-entropy* is defined as

$$H_{\infty}(\mathbf{Z}) = - \min_{x \in \{0,1\}^n} \log p_{\mathbf{Z}}(x).$$

If  $\mathbf{Z}$  is  $\delta$ -dense inside of  $\mathbf{U}$ , then its min-entropy is  $n - \log(1/\delta)$  and for any random variable  $\mathbf{Z}$ ,  $H_{\infty}(\mathbf{Z}) \leq H(\mathbf{Z})$ .

By this latter inequality and subadditivity, the average entropy of  $\mathbf{Z}$ 's bits is at least  $1 - \log(1/\delta)/n$ . Appealing to a quadratic approximation of binary entropy, we learn that the bias must be at most  $\sqrt{\log(1/\delta)/n}$ . This result has been referred to as *Chang's inequality* and the *Level-1 inequality*, having been observed in different forms and with different proofs in, for example, [47, 11, 22, 31]. Because it is so simple, we provide a proof here:

▷ **Claim 3.** If  $\mathbf{Z}$  is  $\delta$ -dense in  $\mathbf{U}$ , then  $\mathbb{E}_i[\text{bias}_i(\mathbf{Z})] \leq \sqrt{\log(1/\delta)/n}$ .

Proof. As  $\delta$ -density is equivalent to  $n - \log(1/\delta)$  min-entropy,

$$n - \log(1/\delta) = H_{\infty}(\mathbf{Z}) \leq H(\mathbf{Z}) \leq \sum_{i \in [n]} H(\mathbf{Z}_i),$$

by subadditivity of entropy. The entropy of  $\mathbf{Z}_i$ 's bits, therefore, is at least  $1 - \log(1/\delta)/n$  on average. Taking the Taylor series, we can approximate the binary entropy function  $h(p)$  around  $1/2$  by a quadratic function as  $h(1/2 + \varepsilon) \leq 1 - (2/\ln 2)\varepsilon^2$ . Comparing this bound with the average, we get

$$1 - \log(1/\delta) \leq 1 - (2/\ln 2)\varepsilon^2,$$

meaning  $\varepsilon \leq \sqrt{(\ln 2/2) \cdot (\log(1/\delta)/n)} \leq \sqrt{\log(1/\delta)/n}$ . ◁

## 2.3 Random variables lacking computational entropy

It follows directly from Claim 3 that if  $\text{bias}_i(\mathbf{Z}_i)$  exceeds  $\varepsilon + \sqrt{\log(1/\delta)/n}$  for every  $i$ , then  $\mathbf{Z}$  does not have a  $\delta$ -dense  $\varepsilon$ -model with respect to the projections  $\mathcal{B}$ .

► **Lemma 4.** Let  $\mathbf{Z}$  be a random variable with  $\text{bias}_i(\mathbf{Z}) \leq \gamma$  for every  $i \in [n]$ . Then for any  $\delta > 0$  and  $\gamma \geq \varepsilon + \sqrt{\frac{\log(1/\delta)}{n}}$ ,  $\mathbf{Z}$  does not have a  $\delta$ -dense  $\varepsilon$ -model with respect to  $\mathcal{B}$ .

This is used for the separation in Theorem 1.3. We would also like a necessary condition for being dense in a pseudorandom set. Towards this end, we note that pseudorandom distributions for even very simple test classes have mild concentration properties.

▷ **Claim 5.** Suppose  $\mathcal{F}$  can compute  $x_i \oplus x_j$  for every  $i, j \in [n]$  and let  $\mathbf{Z}$  over  $\{0,1\}^n$  be  $\varepsilon$ -pseudorandom for  $\mathcal{F}$ . Then

$$\Pr \left[ \sum_i \mathbf{Z}_i \leq n/2 - \alpha n \right] \leq \frac{1}{4\alpha^2 n} + \frac{\varepsilon}{4\alpha^2}.$$

## 2:12 Comparing Computational Entropies Below Majority

*Proof.* We work over  $\{\pm 1\}$  instead of  $\{0, 1\}$  to make calculations easier. We can compute the second moment as

$$\mathbb{E}[(\sum_i \mathbf{Z}_i)^2] = \sum_i \mathbb{E}[\mathbf{Z}_i^2] + \sum_{i \neq j} \mathbb{E}[\mathbf{Z}_i \mathbf{Z}_j] \leq n + \varepsilon n^2.$$

Applying Markov's inequality to  $(\sum_i \mathbf{Z}_i)^2$ , we see that

$$\Pr \left[ \left| \sum_i \mathbf{Z}_i \right| \geq 2\alpha n \right] = \Pr \left[ (\sum_i \mathbf{Z}_i)^2 \geq (2\alpha n)^2 \right] \leq \mathbb{E}[(\sum_i \mathbf{Z}_i)^2] / (2\alpha n)^2.$$

We use  $2\alpha n$  because it maps back to  $n/2 - \alpha n$  in  $\{0, 1\}$ . Then the conclusion follows from our second moment calculation and converting back to  $\{0, 1\}$ .  $\triangleleft$

The tails of a dense subset can't be too much larger than the original distribution, by definition of density. This gives us a test for being dense in a pseudorandom set, which we specialize to  $\mathbf{N}_\alpha$ .

► **Lemma 6.** *Let  $\varepsilon, \delta > 0$  be arbitrary. Suppose  $\mathcal{F}$  can compute  $x_i \oplus x_j$  for any  $i, j \in [n]$  and  $\alpha \geq \sqrt{1/(8\delta) \cdot (1/n + \varepsilon)}$ . Then  $\mathbf{N}_\alpha$  is not  $\delta$ -dense in any set which is  $\varepsilon$ -pseudorandom for  $\mathcal{F}$ .*

**Proof.** Under  $\mathbf{N}_\alpha$ , the volume of the threshold  $\mathbf{1}[\sum_i \mathbf{Z}_i \leq n/2 - \alpha n]$  is  $1/2$ . Taking Claim 5 in the contrapositive, we reach the desired conclusion when

$$\begin{aligned} 1/2 &> \frac{1}{4\delta\alpha^2 n} + \frac{\varepsilon}{4\delta\alpha^2} \\ \alpha^2 &> \frac{1}{8\delta}(1/n + \varepsilon). \end{aligned} \quad \blacktriangleleft$$

## 2.4 Random restrictions and the switching lemma

A restriction over  $[n]$  is a function  $\rho : [n] \rightarrow \{0, 1, *\}$ . Indices in  $\rho^{-1}(*)$  can be thought of as *unset* and each other index as *set*. For another restriction  $z$  so that  $\rho^{-1}(*) \subseteq z^{-1}(\{0, 1\})$ , let  $\rho \circ z \in \{0, 1\}^n$  be defined by

$$(\rho \circ z)_i = \begin{cases} z_i & \text{if } i \in \rho^{-1}(*) \\ \rho_i & \text{otherwise.} \end{cases}$$

Define the restricted function  $f|_\rho : \{0, 1\}^{\rho^{-1}(*)} \rightarrow \{0, 1\}$  over  $\rho$ 's unset indices by

$$f|_\rho(z) = f(\rho \circ z).$$

Let  $R_p$  be the distribution on restrictions over  $[n]$  obtained by setting  $\rho(i) = *$  independently with probability  $p$ , and then setting each bit not assigned to  $*$  a random bit. The switching lemma we use is due to Rossman [39], building on a long line of work [3, 23, 24, 30]:

► **Theorem 2.1** (Rossman [39]). *Suppose  $f \in \mathcal{C}(S, d)$ . Then*

$$\Pr_{\rho \sim R_p} [DT(f|_\rho) \geq k] \leq (p \cdot O(\log S)^{d-1})^k.$$



By considering a random restriction  $\rho \sim R_p$  over  $[n]$  and a random variable  $\mathbf{Z}$  over  $\{0, 1\}^n$ , the definition of a restricted function implies that

$$\mathbb{E}[f(\rho \circ \mathbf{Z})] = \mathbb{E}[f|_{\rho}(\mathbf{Z})].$$

We make crucial use of two simple corollaries of the switching lemma, which allow us to reason about distinguishability for  $\text{AC}^0$  circuits in terms of distinguishability for short decision trees.

► **Lemma 7.** *Suppose  $f \in \mathcal{C}(S, d)$ . Then there is a distribution over depth  $k$  decision trees so that*

$$|\mathbb{E}[f(\rho \circ \mathbf{Z})] - \mathbb{E}[h_{\rho}(\mathbf{Z})]| \leq (p \cdot O(\log S)^{d-1})^k.$$

**Proof.** Let  $g_{\rho}$  denote the optimal decision tree for  $f|_{\rho}$ . Let  $E$  denote the event that  $g_{\rho}$  has depth at most  $k$  and  $\Pr[E] = 1 - q$ . Let  $h_{\rho}$  be the distribution over depth at most  $k$  decision trees obtained by sampling  $g_{\rho}$  conditioned on  $E$ . Then

$$\begin{aligned} \mathbb{E}[f(\rho \circ \mathbf{Z})] &= \mathbb{E}[f|_{\rho}(\mathbf{Z})] \\ &= (1 - q)\mathbb{E}[g_{\rho}(\mathbf{Z})|E] + q\mathbb{E}[g_{\rho}(\mathbf{Z})|\neg E] \\ &= (1 - q)\mathbb{E}[h_{\rho}(\mathbf{Z})] + q\mathbb{E}[g_{\rho}(\mathbf{Z})|\neg E] \\ &= \mathbb{E}[h_{\rho}(\mathbf{Z})] - q(\mathbb{E}[h_{\rho}(\mathbf{Z})] - \mathbb{E}[g_{\rho}(\mathbf{Z})|\neg E]). \end{aligned}$$

The right-hand term is bounded in absolute value by  $q$  because  $f$  is Boolean. By Theorem 2.1,  $q \leq (p \cdot O(\log S)^{d-1})^k$ . ◀

► **Lemma 8.** *Suppose  $f \in \mathcal{C}(S, d)$ . Then there's a depth  $k$  decision tree  $h$  so that*

$$|\mathbb{E}[f(\mathbf{U})] - \mathbb{E}[f(\rho \circ \mathbf{Z})]| \leq |\mathbb{E}[f'(\mathbf{U})] - \mathbb{E}[f'(\mathbf{Z})]| + (p \cdot O(\log S)^{d-1})^k.$$

**Proof.** Lemma 7 gives us the following upper bound.

$$\begin{aligned} |\mathbb{E}[f(\mathbf{U})] - \mathbb{E}[f(\rho \circ \mathbf{Z})]| &\leq |(\mathbb{E}[h_{\rho}(\mathbf{U})] \pm q) - (\mathbb{E}[h_{\rho}(\mathbf{Z})] \pm q)| && \text{(Lemma 7)} \\ &\leq |\mathbb{E}[h_{\rho}(\mathbf{U})] - \mathbb{E}[h_{\rho}(\mathbf{Z})]| + 2q. && \text{(triangle inequality)} \end{aligned}$$

We can continue to upper bound the right-hand term by

$$\begin{aligned} |\mathbb{E}[h_{\rho}(\mathbf{U})] - \mathbb{E}[h_{\rho}(\mathbf{Z})]| &= |\mathbb{E}_{\rho}[\mathbb{E}[h_{\rho}(\mathbf{U})] - \mathbb{E}[h_{\rho}(\mathbf{Z})]]| \\ &\leq \mathbb{E}_{\rho}[|\mathbb{E}[h_{\rho}(\mathbf{U})] - \mathbb{E}[h_{\rho}(\mathbf{Z})]|] && \text{(triangle inequality)} \\ &\leq |\mathbb{E}[h(\mathbf{U})] - \mathbb{E}[h(\mathbf{Z})]| \end{aligned}$$

where the last line holds for some  $h$  in the support of  $h_{\rho}$  by averaging. ◀

### 3 Proof of Theorem 1.3

We start by reducing the problem of constructing a pseudorandom  $\mathbf{Z}$  for  $\text{AC}^0$  to constructing a pseudorandom  $\mathbf{Z}$  for small-depth decision trees. This can be immediately achieved by applying Lemma 8.

▷ **Claim 9.** Let  $p \in [0, 1]$  be arbitrary and suppose  $\mathbf{Z}$  is a random variable over  $\{0, 1\}^n$  which is  $\epsilon$ -pseudorandom for depth- $k$  decision trees. Then for  $\rho \sim R_p$ ,  $\rho \circ \mathbf{Z}$  is  $\epsilon'$ -pseudorandom for  $\mathcal{C}(S, d)$  for

$$\epsilon' = \epsilon + (p \cdot O(\log S)^{d-1})^k.$$

The next lemma constructs a pseudorandom distribution for depth- $k$  decision trees with each bit having significant bias.

► **Lemma 10.** *For any  $k \in \mathbb{N}$ ,  $\delta > 0$  and  $K \geq 1/2\delta$ , there is a  $k$ -wise independent random variable  $\mathbf{S}$  over  $\{0, 1\}^n$  and a  $\delta$ -dense subset  $\mathbf{D}$  of  $\mathbf{S}$  with the property that*

1.  $\mathbf{D}$  is  $\delta$ -dense in  $\mathbf{S}$ .
2. For all  $i \in [n]$ ,  $\text{bias}_i(\mathbf{D}) = \Omega(\sqrt{\log(1/\delta)/8K})$ .
3.  $\mathbf{S}$  is  $k^2/K$ -pseudorandom for depth- $k$  decision trees.

We will use the following standard lower bound on the lower tail of a binomial distribution:

▷ **Claim 11** ([6]). For  $0 < \alpha < 1$  and let  $\mathbf{Z}_1, \dots, \mathbf{Z}_K$  be independent unbiased coins ( $\{0, 1\}$ -valued). Then any  $\gamma$  with  $1/2 - \gamma = r/K$  for some positive integer  $r$  satisfies

$$\frac{2^{-K(1-h(1/2-\gamma))}}{\sqrt{2K}} \leq \Pr \left[ \sum_{i \in [K]} \mathbf{Z}_i \leq K/2 - K\gamma \right].$$

**Proof of Lemma 10.** We sample  $\mathbf{S}$  in two stages. First, randomly partition  $[n]$  into  $K$  parts  $\mathbf{A}_1, \dots, \mathbf{A}_K$  for  $K > k^2$ . Second, assign to each  $A_i$  a uniformly random bit  $\mathbf{b}_i$ .

Let  $\mathbf{D}$  be  $\mathbf{S}$  conditioned on  $\mathbf{b} = (\mathbf{b}_1, \dots, \mathbf{b}_K)$  having weight less than  $K/2 - \sqrt{K \log(1/\delta)/8}$ . Since the  $\mathbf{b}_i$ 's are unbiased random bits, we can apply Claim 11 to lower bound  $\mathbf{D}$ 's density: for any  $\gamma$ ,

$$\Pr \left[ \sum_{i \in [k]} \mathbf{b}_i \leq \gamma K \right] \geq \frac{2^{-Kh(1/2-\gamma)}}{\sqrt{2K}}.$$

This is at least  $\delta$  when

$$\begin{aligned} \frac{2^{-K(1-h(1/2-\gamma))}}{\sqrt{2K}} &\geq \delta \\ 1 - h(1/2 - \gamma) &\geq \log(1/\delta)/K - \log(2K)/2K \\ 4\gamma^2 &\geq \log(1/\delta)/K - \log(2K)/2K \end{aligned}$$

with the upper bound in the last line following from  $h(1/2 - \gamma) \geq 1 - 4\gamma^2$ . Hence, if the set of strings with weight at most  $K/2 - \gamma K$  is  $\delta$ -dense, we have  $\gamma \geq \frac{1}{2} \sqrt{\log(1/\delta)/K - \log(2K)/2K}$ .  $\log(2K)/2K$  is at most  $\log(1/\delta)/2K$  when  $2K \leq 1/\delta$ , in which case  $\gamma \geq \sqrt{\log(1/\delta)/8K}$ . In particular, this lower bounds the bias of  $\mathbf{D}$ 's bits.

To see why it's  $k^2/K$ -pseudorandom for depth- $k$  decision trees, consider a depth- $k$  decision tree  $T$ . Over  $\mathbf{U}$ , we can imagine evaluating  $T$  “on-line” as follows: whenever  $T$  queries the  $i$ th bit, determine the value of  $z_i$  by flipping an unbiased coin. Over  $\mathbf{S}$ , we can imagine evaluating  $T$  similarly, where we determine the bucket  $A_j$  that  $i$  lives in and the value  $b_j$  of that bucket.

By conditioning  $\mathbf{S}$  on *not* placing two distinct indices  $i, j$  in the same bucket – call this conditioned random variable  $\mathbf{S}'$  – then  $T$  doesn't have *any* distinguish advantage over  $\mathbf{S}'$ , as all of the bits it queries are independent and uniform. By a union bound,  $\mathbf{S}$  places two distinct indices in the same bucket with probability at most  $k^2/K$ .  $T$ 's distinguishing advantage is therefore at most  $k^2/K$ . ◀

In principle, we could have used other pseudorandom distributions for decision trees such as the  $\varepsilon$ -almost  $k$ -wise independent distributions from [4]. The construction here is used to obtain better dependence on the parameters of interest. We will also need a claim to express the bias of the bits in  $\rho \circ \mathbf{Z}$ . The proof can be found in the full version of the paper.

▷ **Claim 12.** Fix  $p \in [0, 1]$  and a random variable  $\mathbf{Z}$ . Let  $E$  be an event which is independent from  $\rho$  (in that the conditional distribution of  $\rho$  is identical to the unconditioned distribution). Then

$$\Pr[(\rho \circ \mathbf{Z})_i = 1 | E] = p \Pr[\mathbf{Z}_i = 1 | E] + (1 - p)/2.$$

Theorem 1.3, which we restate here, is obtained by an appropriate setting of parameters.

► **Theorem 1.3.** Let  $\varepsilon, \varepsilon' > 0$  be arbitrary,  $\delta \geq \varepsilon'/8$  and

$$S \leq \exp\left(O\left(\frac{\sqrt{\varepsilon'}}{\varepsilon} \cdot \frac{\sqrt{\log(1/\delta)}}{\log(1/\varepsilon')}\right)^{1/(d-1)}\right).$$

Then for  $\mathcal{F} = \mathcal{C}(S, d)$ , there is a random variable  $\mathbf{D}$  over  $\{0, 1\}^n$  with  $n = O(\log(1/\delta)/\varepsilon^2)$  so that  $\mathbf{D}$  is  $\delta$ -dense in an  $\varepsilon'$ -pseudorandom set but does not have a  $\delta$ -dense  $\varepsilon$ -model. In particular, the dense model theorem is false in this setting.

**Proof.** Let  $n = \log(1/\delta)/\varepsilon^2$ ,  $k = \log(2/\varepsilon')$  and  $K = (2k^2)/\varepsilon'$ . We also need  $K \geq 1/2\delta$  by the restriction in Lemma 10, which explains the restriction  $8\delta k^2 \geq \varepsilon'$ , simplified by using  $8\delta \geq \varepsilon'$  (a stronger restriction) instead. Let  $\mathbf{S}$  and  $\mathbf{D}$  be the random variables from Lemma 10. By Claim 12, the bias of  $\rho \circ \mathbf{S}$  (where  $\rho \sim R_p$ ) is  $p\sqrt{\log(1/\delta)/8K}$ . By Claim 9 and Lemma 10,  $\rho \circ \mathbf{S}$  is  $\varepsilon' = k^2/K + (pO(\log S)^{d-1})^k$  pseudorandom. We can also ensure that  $\rho \circ \mathbf{S}$  does not have a  $\delta$ -dense  $\varepsilon$ -model when  $p\sqrt{\log(1/\delta)/8K} \geq \varepsilon + \sqrt{\log(1/\delta)/n}$ , by Lemma 4.

By substituting,  $p \geq 2\sqrt{K/n} = 2\sqrt{(k\varepsilon)^2/\varepsilon' \cdot \log(1/\delta)}$ . In comparison,  $\varepsilon' \geq k^2/K + (pO(\log S)^{d-1})^k$ . Recalling that  $k^2/K = \varepsilon'/2$ , we get that

$$\begin{aligned} \varepsilon'/2 &\geq (2\sqrt{K/n}O(\log S)^{d-1})^k \\ \frac{\sqrt{n}}{2\sqrt{K}}(\varepsilon'/2)^{1/k} &\geq O(\log S)^{d-1} \\ \frac{\sqrt{\log 1/\delta}}{\varepsilon} \cdot \frac{\sqrt{\varepsilon'}}{2\sqrt{2k}} \cdot (\varepsilon'/2)^{1/k} &\geq O(\log S)^{d-1} \\ \frac{\sqrt{\log 1/\delta}}{\varepsilon} \cdot \frac{\sqrt{\varepsilon'}}{\sqrt{32\log(1/\varepsilon')}} &\geq O(\log S)^{d-1}. \end{aligned}$$

The claim follows by solving for  $S$ . ◀

## 4 Proof of Theorem 1.4

Theorem 1.4 follows by combining Lemma 6 and the following lemma:

► **Lemma 13.**  $\mathbf{N}_\alpha$  has  $(\varepsilon, \delta)$ -pseudodensity for  $\mathcal{C}(S, d)$  for  $\varepsilon = (p \cdot O(\log S)^{d-1})^k$  and  $\delta = e^{-\alpha k/p}$ .

Of note, the only additive error depends on the error from the switching lemma. Compare this with the claim that  $\mathbf{N}_\alpha$  is  $(3\alpha \cdot O(\log S)^{d-1})$ -pseudorandom (and therefore has the same pseudodensity for  $\delta = 1$ ) for  $\mathcal{C}(S, d)$ , due to Tal [46].

To prove the lemma, we need a few claims.

## 2:16 Comparing Computational Entropies Below Majority

▷ **Claim 14.** Suppose  $f \in \mathcal{C}(S, d)$ . Then there is a depth- $k$  decision tree  $h$  with the property that:

$$\mathbb{E}[f(\mathbf{N}_\alpha)] \leq \mathbb{E}[h(\mathbf{N}_{\alpha/p})] + (p \cdot O(\log S)^{d-1})^k.$$

**Proof.** Take  $\mathbf{Z} = \mathbf{N}_{\alpha/p}$  in Lemma 7, so we have  $\rho \circ \mathbf{N}_{\alpha/p} = \mathbf{N}_\alpha$  and

$$\mathbb{E}[f(\mathbf{N}_\alpha)] \leq \mathbb{E}[h_\rho(\mathbf{N}_{\alpha/p})] + (p \cdot O(\log S)^{d-1})^k.$$

Averaging over  $\rho$  yields the fixed decision tree. ◁

Second, we can upper bound the extent to which the acceptance probability of a short decision tree increases when passing from the uniform distribution  $\mathbf{U}$  to the biased distribution  $\mathbf{N}_\gamma$ .

▷ **Claim 15.** Suppose  $f : \{0, 1\}^n \rightarrow \{-1, 1\}$  is a depth- $k$  decision tree. Then

$$\mathbb{E}[f(\mathbf{N}_\gamma)] \leq (1 + \gamma)^k \cdot \mathbb{E}[f(\mathbf{U})] \leq e^{\gamma k} \cdot \mathbb{E}[f(\mathbf{U})].$$

The proof is simple and can be found in the full version. We're now in a position to prove the lemma.

**Proof of Lemma 13.** Directly applying Claim 14, we get

$$\mathbb{E}[f(\mathbf{N}_\alpha)] \leq \mathbb{E}[f'(\mathbf{N}_{\alpha/p})] + (p \cdot O(\log S)^{d-1})^k.$$

Applying Claim 15 to  $\mathbb{E}[f'(\mathbf{N}_{\alpha/p})]$ , we get

$$\begin{aligned} \mathbb{E}[f(\mathbf{N}_\alpha)] &\leq (1 + \alpha/p)^k \mathbb{E}[f'(\mathbf{U})] \\ &\leq e^{\alpha k/p} \mathbb{E}[f'(\mathbf{U})]. \end{aligned}$$

Putting these together finishes the proof. ◀

We can now prove Theorem 1.4, restated here:

► **Theorem 1.4.** Fix  $\varepsilon, \varepsilon', \delta > 0$ ,  $d \in \mathbb{N}$ , and

$$S \leq \exp \left( O \left( \frac{\sqrt{\delta}}{\sqrt{\varepsilon}} \cdot \frac{\log(1/\delta)}{\log(1/\varepsilon')} \right)^{1/(d-1)} \right).$$

Then  $\mathbf{N}_{\sqrt{\varepsilon/\delta}}$  over  $\{0, 1\}^n$  with  $n = O(1/\varepsilon)$  is  $(\varepsilon', \delta)$ -pseudodense and yet  $\mathbf{N}_{\sqrt{\varepsilon/\delta}}$  is not  $\delta$ -dense inside of any  $\varepsilon$ -pseudorandom set.

**Proof of Theorem 1.4.** Let  $n = 1/(7\varepsilon)$ ,  $k = \log(1/\varepsilon')$  and  $\alpha = \sqrt{\varepsilon/\delta}$ . These choices satisfy  $\alpha \geq \sqrt{\frac{1}{8\delta}}(1/n + \varepsilon)$ , meaning  $\mathbf{N}_\alpha$  is not  $\delta$ -dense in any  $\varepsilon$ -pseudorandom set for  $\mathcal{C}(S, d)$ , by Lemma 6.

By Lemma 13,  $\mathbf{N}_\alpha$  has  $(\varepsilon', \delta)$ -pseudodensity for  $\delta = e^{-\alpha k/p}$  and  $\varepsilon' = (p \cdot O(\log S)^{d-1})^k$ . The constraint on the density implies

$$\delta = e^{-\alpha k/p}$$

$$\log(1/\delta) = \alpha k/p$$

$$\log(1/\delta) = \sqrt{\varepsilon/\delta} \log(1/\varepsilon')/p$$

$$p = \frac{\sqrt{\varepsilon/\delta} \log(1/\varepsilon')}{\log(1/\delta)}.$$

Plugging this value of  $p$  into the expression for  $\varepsilon'$ , we get

$$\begin{aligned}\varepsilon' &= (p \cdot O(\log S)^{d-1})^k \\ (\varepsilon')^{1/k}/p &= O(\log S)^{d-1} \\ (\varepsilon')^{1/\log(1/\varepsilon')} \cdot \frac{\sqrt{\delta} \log(1/\delta)}{\sqrt{\varepsilon} \log(1/\varepsilon')} &= O(\log S)^{d-1}.\end{aligned}$$

Note that  $(\varepsilon')^{1/\log(1/\varepsilon')} = 2^{-\log(1/\varepsilon')/\log(1/\varepsilon')} = 1/2$ . Solving for  $S$  gives the claimed bound.  $\blacktriangleleft$

## 5 Discussion of Theorem 1.5 and Theorem 1.6

This section briefly discusses Theorem 1.5 and Theorem 1.6, deferring a more detailed discussion to the full version of the paper. The basic idea underlying both proofs is to use tests which solve the coin problem to construct a test which “computes majority” in some problem-dependent sense.

Theorem 1.5 shows that the dense model theorem can fail for low-degree polynomials over finite fields.

► **Theorem 1.5.** *Fix a finite field  $\mathbb{F}$  with characteristic  $p = O(1)$ ,  $\varepsilon, \varepsilon' > 0$  and let  $c > \delta > 0$  where  $c \approx 1/200$  is an absolute constant. Suppose that*

$$d \leq O(\sqrt{\delta/\varepsilon}).$$

*Then when  $\mathcal{F}$  is the  $n$ -variate degree- $d$  polynomials over  $\mathbb{F}$  with  $n = 1/\varepsilon$ , and  $\alpha = O(\sqrt{\varepsilon/\delta})$ ,  $\mathbf{N}_\alpha$  is  $(\varepsilon', \delta)$ -pseudodense but is not  $\delta$ -dense inside of an  $\varepsilon$ -pseudorandom set.*

The main tool used in the proof is a special case of the robust Heg  dus lemma, discovered recently by Srinivasan [44].

► **Lemma 16** (Robust Heg  dus lemma (special case), [44]). *Let  $\mathbb{F}$  be a finite field. Let  $2^{-m/100} \leq \lambda \leq c$  where  $c < 1$  is a (small) absolute constant. Let  $\alpha^2 m$  be an integer so that  $2^{-2\alpha^2 m} \geq \lambda$ . Then if  $P : \mathbb{F}^n \rightarrow \mathbb{F}$  is a degree  $d$  polynomial for which:*

1.  $\Pr[P(\mathbf{Sp}_{m, \alpha m}) \neq 0] \leq \lambda$
2.  $\Pr[P(\mathbf{Sp}_{m, 0}) = 0] \leq 1 - e^{-\alpha^2 m/2}$ .

*Then  $d = \Omega(\alpha m)$ .*

The idea is to use a low-degree polynomial distinguishing the biased coin distribution from uniform to construct another low-degree polynomial satisfying the conditions in the above lemma. Our particular approach uses random sampling and the approximation of OR by low-degree probabilistic polynomials [37]. We defer the details to the full version.

Theorem 1.6 gives a generic condition under which the dense model theorem is false, being witnessed by biased coins.

► **Theorem 1.6.** *Let  $\varepsilon, \delta > 0$ . Suppose  $\mathcal{F}$  is a test class of boolean functions  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  with the following property: there is no  $\text{AC}^0$   $\mathcal{F}$ -oracle circuit of size  $\text{poly}(n \cdot \frac{\sqrt{\delta}}{\varepsilon^{3/2}})$  computing majority on  $O(\sqrt{\delta/\varepsilon})$  bits.*

*Then  $\mathbf{N}_{\sqrt{\varepsilon/\delta}}$  is  $(\varepsilon\delta, \delta)$ -pseudodense and yet does not have a  $\delta$ -dense  $\varepsilon$ -model. In particular, when the hypotheses are met, the dense model theorem is false.*

The approach is the same as Theorem 1.5, this time using a small circuit distinguishing biased from uniform to build an only-slightly-larger circuit computing majority. In this case, we use the following result of Shaltiel & Viola:

► **Theorem 5.1** ([42]). *Let  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  be a function that distinguishes between  $\mathbf{U}$  and  $\mathbf{N}_\alpha$  with constant distinguishing probability. Then there is an  $\text{AC}^0$ -circuit of size  $\text{poly}(n/\alpha)$  using  $f$ -oracle gates which computes majority on  $O(1/\alpha)$  bits.*

Once again, we defer the details to the full version.

---

## References

- 1 Scott Aaronson. Bqp and the polynomial hierarchy. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 141–150, 2010.
- 2 Rohit Agrawal. Coin theorems and the fourier expansion. *arXiv preprint*, 2019. [arXiv:1906.03743](#).
- 3 Miklos Ajtai and Avi Wigderson. Deterministic simulation of probabilistic constant depth circuits. In *26th Annual Symposium on Foundations of Computer Science (sfcs 1985)*, pages 11–19. IEEE, 1985.
- 4 Noga Alon, Oded Goldreich, Johan Håstad, and René Peralta. Simple constructions of almost  $k$ -wise independent random variables. *Random Structures & Algorithms*, 3(3):289–304, 1992.
- 5 Sanjeev Arora, Rong Ge, Yingyu Liang, Tengyu Ma, and Yi Zhang. Generalization and equilibrium in generative adversarial nets (gans). In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 224–232. JMLR. org, 2017.
- 6 R.B. Ash. *Information Theory*. Dover books on advanced mathematics. Dover Publications, 1990. URL: <https://books.google.com/books?id=nJ3UmGvdUCoC>.
- 7 Boaz Barak, Moritz Hardt, and Satyen Kale. The uniform hardcore lemma via approximate bregman projections. In *Proceedings of the twentieth annual ACM-SIAM symposium on Discrete algorithms*, pages 1193–1200. SIAM, 2009.
- 8 Boaz Barak, Ronen Shaltiel, and Avi Wigderson. Computational analogues of entropy. In *Approximation, Randomization, and Combinatorial Optimization.. Algorithms and Techniques*, pages 200–215. Springer, 2003.
- 9 Thomas F Bloom and Olof Sisask. Breaking the logarithmic barrier in roth’s theorem on arithmetic progressions. *arXiv preprint*, 2020. [arXiv:2007.03528](#).
- 10 Joshua Brody and Elad Verbin. The coin problem and pseudorandomness for branching programs. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 30–39. IEEE, 2010.
- 11 Mei-Chu Chang et al. A polynomial bound in freiman’s theorem. *Duke mathematical journal*, 113(3):399–419, 2002.
- 12 Gil Cohen, Anat Ganor, and Ran Raz. Two sides of the coin problem. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2014)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2014.
- 13 Dean Doron, Dana Moshkovitz, Justin Oh, and David Zuckerman. Nearly optimal pseudorandomness from hardness. Technical report, ECCC preprint TR19-099, 2019.
- 14 Stefan Dziembowski and Krzysztof Pietrzak. Leakage-resilient cryptography. In *2008 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 293–302. IEEE, 2008.
- 15 Yoav Freund and Robert Schapire. A short introduction to boosting. *Journal-Japanese Society For Artificial Intelligence*, 14(771-780):1612, 1999.
- 16 W. T. Gowers. A new proof of szemerédi’s theorem. *Geometric & Functional Analysis GAFA*, 11(3):465–588, 2001.
- 17 W. T. Gowers. Decompositions, approximate structure, transference, and the hahn–banach theorem. *Bulletin of the London Mathematical Society*, 42(4):573–606, 2010.

- 18 Ben Green and Terence Tao. The primes contain arbitrarily long arithmetic progressions. *Annals of Mathematics*, pages 481–547, 2008.
- 19 Aryeh Grinberg, Ronen Shaltiel, and Emanuele Viola. Indistinguishability by adaptive procedures with advice, and lower bounds on hardness amplification proofs. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 956–966. IEEE, 2018.
- 20 Iftach Haitner, Omer Reingold, and Salil Vadhan. Efficiency improvements in constructing pseudorandom generators from one-way functions. *SIAM Journal on Computing*, 42(3):1405–1430, 2013.
- 21 Iftach Haitner, Omer Reingold, Salil Vadhan, and Hoeteck Wee. Inaccessible entropy. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 611–620, 2009.
- 22 Lianna Hambardzumyan and Yaqiao Li. Chang’s lemma via pinsker’s inequality. *Discrete Mathematics*, 343(1):111496, 2020.
- 23 Johan Håstad. *Computational limitations of small-depth circuits*. MIT Press, 1987.
- 24 Johan Håstad. On the correlation of parity and small-depth circuits. *SIAM Journal on Computing*, 43(5):1699–1708, 2014.
- 25 Johan Håstad, Russell Impagliazzo, Leonid A Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999.
- 26 Thomas Holenstein. Key agreement from weak bit agreement. In *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 664–673, 2005.
- 27 Chun-Yuan Hsiao, Chi-Jen Lu, and Leonid Reyzin. Conditional computational entropy, or toward separating pseudoentropy from compressibility. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 169–186. Springer, 2007.
- 28 Russell Impagliazzo. Hard-core distributions for somewhat hard problems. In *Proceedings of IEEE 36th Annual Foundations of Computer Science*, pages 538–545. IEEE, 1995.
- 29 Russell Impagliazzo. Connections between pseudo-randomness and machine learning: boosting, dense models, and regularity, 2020.
- 30 Russell Impagliazzo, William Matthews, and Ramamohan Paturi. A satisfiability algorithm for  $ac_0$ . In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*, pages 961–972. SIAM, 2012.
- 31 Russell Impagliazzo, Cristopher Moore, and Alexander Russell. An entropic proof of chang’s inequality. *SIAM Journal on Discrete Mathematics*, 28(1):173–176, 2014.
- 32 Adam R Klivans and Rocco A Servedio. Boosting and hard-core set construction. *Machine Learning*, 51(3):217–238, 2003.
- 33 Nutan Limaye, Karteeek Sreenivasaiiah, Srikanth Srinivasan, Utkarsh Tripathi, and S Venkitesh. A fixed-depth size-hierarchy theorem for  $AC^0[\oplus]$  via the coin problem. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 442–453, 2019.
- 34 Chi-Jen Lu, Shi-Chun Tsai, and Hsin-Lung Wu. Complexity of hard-core set proofs. *computational complexity*, 20(1):145–171, 2011.
- 35 Or Meir and Avi Wigderson. Prediction from partial information and hindsight, with application to circuit lower bounds. *computational complexity*, 28(2):145–183, 2019.
- 36 Ilya Mironov, Omkant Pandey, Omer Reingod, and Salil Vadhan. Computational differential privacy. In *Annual International Cryptology Conference*, pages 126–142. Springer, 2009.
- 37 Alexander A Razborov. Lower bounds on the size of bounded depth circuits over a complete basis with logical addition. *Mathematical Notes of the Academy of Sciences of the USSR*, 41(4):333–338, 1987.
- 38 Omer Reingold, Luca Trevisan, Madhur Tulsiani, and Salil Vadhan. Dense subsets of pseudorandom sets. In *2008 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 76–85. IEEE, 2008.
- 39 Benjamin Rossman. An entropy proof of the switching lemma and tight bounds on the decision-tree size of  $ac_0$ , 2017.

- 40 Rocco A Servedio and Li-Yang Tan. Improved pseudorandom generators from pseudorandom multi-switching lemmas. *arXiv preprint*, 2018. [arXiv:1801.03590](#).
- 41 Ronen Shaltiel. Is it possible to improve yao’s xor lemma using reductions that exploit the efficiency of their oracle? In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.
- 42 Ronen Shaltiel and Emanuele Viola. Hardness amplification proofs require majority. *SIAM Journal on Computing*, 39(7):3122–3154, 2010.
- 43 Roman Smolensky. On representations by low-degree polynomials. In *Proceedings of 1993 IEEE 34th Annual Foundations of Computer Science*, pages 130–138. IEEE, 1993.
- 44 Srikanth Srinivasan. A robust version of hegedus’s lemma, with applications. In Konstantin Makarychev, Yury Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy, editors, *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*, pages 1349–1362. ACM, 2020. doi: 10.1145/3357713.3384328.
- 45 Endre Szemerédi. On sets of integers containing no four elements in arithmetic progression. *Acta Mathematica Academiae Scientiarum Hungarica*, 20(1-2):89–104, 1969.
- 46 Avishay Tal. Tight bounds on the fourier spectrum of ac0. In *32nd Computational Complexity Conference (CCC 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.
- 47 Michel Talagrand. How much are increasing sets positively correlated? *Combinatorica*, 16(2):243–258, 1996.
- 48 Terence Tao, Tamar Ziegler, et al. The primes contain arbitrarily long polynomial progressions. *Acta Mathematica*, 201(2):213–305, 2008.
- 49 Luca Trevisan, Madhur Tulsiani, and Salil Vadhan. Regularity, boosting, and efficiently simulating every high-entropy distribution. In *2009 24th Annual IEEE Conference on Computational Complexity*, pages 126–136. IEEE, 2009.
- 50 Thomas Watson. Advice lower bounds for the dense model theorem. *ACM Transactions on Computation Theory (TOCT)*, 7(1):1–18, 2015.
- 51 Andrew C Yao. Theory and application of trapdoor functions. In *23rd Annual Symposium on Foundations of Computer Science (SFCS 1982)*, pages 80–91. IEEE, 1982.
- 52 Jiapeng Zhang. On the query complexity for showing dense model. *Electron. Colloquium Comput. Complex.*, 2011.



# Algorithmic Persuasion with Evidence

Martin Hoefer 

Goethe Universität Frankfurt, Germany  
mhoefer@cs.uni-frankfurt.de

Pasin Manurangsi 

Google Research, Mountain View, CA, USA  
pasin@google.com

Alexandros Psomas

Purdue University, West Lafayette, IN, USA  
apsomas@cs.purdue.edu

---

## Abstract

We consider a game of persuasion with evidence between a sender and a receiver. The sender has private information. By presenting evidence on the information, the sender wishes to persuade the receiver to take a single action (e.g., hire a job candidate, or convict a defendant). The sender's utility depends solely on whether or not the receiver takes the action. The receiver's utility depends on both the action as well as the sender's private information. We study three natural variations. First, we consider sequential equilibria of the game without commitment power. Second, we consider a persuasion variant, where the sender commits to a signaling scheme and then the receiver, after seeing the evidence, takes the action or not. Third, we study a delegation variant, where the receiver first commits to taking the action if being presented certain evidence, and then the sender presents evidence to maximize the probability the action is taken. We study these variants through the computational lens, and give hardness results, optimal approximation algorithms, as well as polynomial-time algorithms for special cases. Among our results is an approximation algorithm that rounds a semidefinite program that might be of independent interest, since, to the best of our knowledge, it is the first such approximation algorithm for a natural problem in algorithmic economics.

**2012 ACM Subject Classification** Theory of computation → Algorithmic game theory and mechanism design; Theory of computation → Approximation algorithms analysis

**Keywords and phrases** Bayesian Persuasion, Semidefinite Programming, Approximation Algorithms

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.3

**Related Version** A full version of the paper is available at <http://arxiv.org/abs/2008.12626>.

**Funding** *Martin Hoefer*: GIF grant I-1419-118.4/2017 and DFG grants Ho 3831/5-1, 6-1, and 7-1.

**Acknowledgements** This work was done in part while Martin Hoefer and Alexandros Psomas were visiting the Simons Institute for the Theory of Computing. The authors acknowledge financial support and the invitation by the organizers to join the stimulating work environment. This work was done in part while Alexandros Psomas was visiting Google Research, Mountain View, USA.

## 1 Introduction

Persuasion is a fundamental challenge arising in diverse areas such as recommendation problems in the Internet, consulting and lobbying, employee hiring, and many more. Persuasion problems occupy a central role in economics and received significant interest over the last two decades. A prominent approach is *persuasion with evidence* as introduced by Glazer and Rubinstein [13, 14], which has attracted a lot of subsequent work. In this problem, a sender wishes to persuade a receiver to take a single action by presenting evidence. The sender's utility depends solely on whether or not the action is taken, while the receiver's utility



© Martin Hoefer, Pasin Manurangsi, and Alexandros Psomas;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 3; pp. 3:1–3:20



Leibniz International Proceedings in Informatics

LIPIcs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

depends on both the action as well as the sender’s private information. Consider, for example, a prosecutor trying to convince a judge that a defendant is guilty and should be convicted, or a job candidate trying to convince a company that she has the best qualifications and should be hired. How should these pairs of agents interact?

The literature on persuasion games in economics and game theory is vast; see Sobel [27] for a survey. In sharp contrast, very little is known about *computation* in this domain, especially for the persuasion problem with evidence. How does the restriction to evidence impact the computational complexity of the problem? Our main contribution of this paper is to initiate the systematic study of *persuasion with evidence through a computational lens*. We examine three natural model variants that arise from the power to commit to certain behavior.

If there is no commitment power, the scenario is an extensive-form game. We prove that finding a sequential equilibrium is always possible in polynomial time. However, the sender and the receiver can significantly improve their utility when they enjoy commitment power.

If the sender has commitment power, then she can commit in advance which evidence is presented in each possible instantiation of her private information, and the receiver seeing the evidence then takes the action or not. We refer to this situation as *constrained persuasion*, since the sender with commitment power wants to persuade the rational receiver to take the action. The sender is constrained to providing concrete evidence instead of just making a recommendation as is the case in the so called Bayesian persuasion paradigm [19]. Constrained persuasion is a natural model in the example of prosecutor and judge, where the prosecutor (sender) with private information would first present evidence before the judge (receiver) makes a decision. Although this scenario seems structurally rather simple, we show that the sender’s task in constrained persuasion is computationally (highly) intractable. Unless  $P = NP$ , optimal persuasion can become hard to approximate within a polynomial factor of the input size.

If the receiver has commitment power, she commits to taking the action if and only if being faced with a specific set of evidence. We refer to this situation as *constrained delegation*, since we assume that the receiver with commitment power delegates inspection of the state of nature to a sender, whose incentive becomes to provide convincing evidence to support taking the action. Constrained delegation better fits the second example, where the company (receiver) can give the candidate (sender) a test to present evidence on the private information about qualifications, and commit to hiring her if she performs well. We show that the receiver’s task in delegation is also intractable – unless  $P = NP$ , optimal delegation can become hard to approximate within a factor of  $2 - \varepsilon$ , for any constant  $\varepsilon > 0$ .

These computational differences nicely reflect conceptual differences known from the economics literature. Namely, persuasion lacks a condition termed “credibility” that was shown for delegation. Formally, credibility implies that there is a deterministic optimal solution that does not require randomization, see Glazer and Rubinstein [14] for details. We proceed to study algorithms with matching approximation guarantees for constrained persuasion and delegation, as well as a number of exact and approximation algorithms for various special cases. This includes, in particular, an approximation algorithm for a class of delegation problems that solves and rounds a semidefinite program (SDP). This last result might be of independent interest and, to the best of our knowledge, it is the first natural problem in information structure design, as well as mechanism design, where the SDP toolbox is used.

## 2 Preliminaries

Following [13, 14, 25], we study the fundamental problem of persuasion with evidence. There are two players, a sender and a receiver. The receiver is tasked with either taking a specific action and “accept” (henceforth  $A$ ), or sticking to the status quo and “reject” (henceforth  $R$ ). The sender wants to convince the receiver to take action  $A$ . There is a state of nature  $\theta$  drawn from a distribution  $\mathcal{D}$  with support  $\Theta$  of size  $n$ . We denote the probability that  $\theta$  is drawn by  $q_\theta$ . The set  $\Theta$  is partitioned into the set of acceptable states  $\Theta_A$  and the set of rejectable ones  $\Theta_R = \Theta \setminus \Theta_A$ . We denote the total probability on acceptable states by  $q_A = \sum_{\theta \in \Theta_A} q_\theta$ , and the total probability on rejectable states by  $q_R = \sum_{\theta \in \Theta_R} q_\theta$ .

Both players know  $\mathcal{D}$ . The sender knows the realization of the state of nature, the receiver does not. The sender has utility 1 whenever the receiver takes action  $A$ , and 0 otherwise. Formally, for the sender utility we have  $u_s(A, \theta) = 1$  and  $u_s(R, \theta) = 0$ , for all  $\theta \in \Theta$ .

The utility of the receiver depends on the combination of the chosen action  $a \in \{A, R\}$  and the state of nature  $\theta$ . She has utility 1 if she makes the “right” decision – accept in an acceptable state of nature or reject in a rejectable state of nature – and 0 otherwise. Formally,  $u_r(a, \theta) = 1$  when (1)  $a = A$  and  $\theta \in \Theta_A$ , or (2)  $a = R$  and  $\theta \in \Theta_R$ . Otherwise,  $u_r(a, \theta) = 0$ .

The sender strives to send a message to the receiver according to a public signaling strategy. This message should persuade the receiver to accept. On the other hand, upon receiving the message, the receiver strives to infer the state of nature and make the right accept/reject decision. We focus on games with evidence, where the messages that can be sent are not arbitrary. Every state of nature has intrinsic characteristics (e.g., a candidate for a position has grades, degrees, or test scores) that *can* be (but don’t *have* to be) revealed to the receiver and cannot be forged.

More formally, there is a set  $\Sigma$  of  $m$  possible messages or *signals* that the sender can report to the receiver. We are given as input a bipartite graph  $H = (\Theta \cup \Sigma, E)$ , where an edge  $e = (\theta, \sigma) \in E$  implies that signal  $\sigma$  is allowed to be sent in state  $\theta$ . We use  $N(\theta) \subseteq \Sigma$  to denote the neighborhood of  $\theta$ , i.e., the set of allowed signals for state  $\theta$ . Similarly,  $N(\sigma) \subseteq \Theta$  is the set of states in which signal  $\sigma$  can be sent. To avoid trivialities, we assume that none of the neighborhoods  $N(\cdot)$  are empty, i.e., there are no isolated nodes in  $H$ .

We study the computational complexity of games with evidence for different forms of interaction between the sender and the receiver. In particular, in the case of *constrained persuasion*, the game starts with the sender committing to a *signaling scheme*. A signaling scheme  $\varphi$  is a mapping  $\varphi : E \rightarrow [0, 1]$ , where  $\varphi(\theta, \sigma)$  is the joint probability that state  $\theta$  is realized and signal  $\sigma$  is sent in state  $\theta$ . Clearly, for any signaling scheme we have  $\sum_{\sigma \in N(\theta)} \varphi(\theta, \sigma) = q_\theta$  for every state  $\theta \in \Theta$ . After the sender has committed to a scheme  $\varphi$ , nature draws  $\theta \in \Theta$  with probability  $q_\theta$ , and  $\theta$  is revealed to the sender. Then, the sender sends signal  $\sigma$  with probability  $\varphi(\theta, \sigma)/q_\theta$ . The receiver then decides on an action  $A$  or  $R$ . Finally, depending on the (state of nature, action)-pair, the sender and receiver get payoffs as described by the utilities above.

► **Problem 1 (CONSTRAINED PERSUASION).** *Find a signaling scheme  $\varphi^*$  for commitment of the sender such that, upon a best response of the receiver, the sender utility is as high as possible.*

In the case of *constrained delegation*, the game starts with the receiver committing to an action for every possible signal  $\sigma \in \Sigma$ , according to a *decision scheme*. A decision scheme  $\psi$  is a mapping  $\psi : \Sigma \rightarrow [0, 1]$ , where  $\psi(\sigma)$  is the probability to choose action  $A$ . After the receiver has committed to a scheme  $\psi$ , nature draws  $\theta \in \Theta$  with probability  $q_\theta$ , and  $\theta$  is revealed to the sender. Then, the sender decides which signal  $\sigma$  she will report (under the

constraint that  $\sigma \in N(\theta)$ ). The receiver then takes action A with probability  $\psi(\sigma)$ , and R otherwise. Finally, depending on the (state of nature, action)-pair, the sender and receiver get payoffs as described by the utilities above.

► **Problem 2 (CONSTRAINED DELEGATION).** *Find a decision scheme  $\psi^*$  for commitment of the receiver such that, upon a best response of the sender, the receiver utility is as high as possible.*

Finally, in the game without commitment power, we look for a pair  $(\varphi, \psi)$  of signaling and decision schemes that constitute a sequential equilibrium in the extensive-form game, where nature first determines the state of nature, the sender then picks  $\varphi$  to provide evidence, and then the receiver uses  $\psi$  to accept or reject based on the evidence provided. Given that the sender picks  $\varphi$ , the receiver shall pick  $\psi$  as a best response for every given evidence. Similarly, given that the receiver responds to evidence with  $\psi$ , the signaling scheme  $\varphi$  shall be a best response for the sender.

► **Problem 3 (CONSTRAINED EQUILIBRIUM).** *Find a pair of signaling scheme  $\varphi$  and decision scheme  $\psi$  that represents a sequential equilibrium in the persuasion game with evidence and without commitment power.*

## 2.1 Structural Properties

While the persuasion problem with evidence appears rather elementary, it turns out that both persuasion and delegation variants are NP-hard, and even NP-hard to approximate in polynomial time. Hence, even in this seemingly simple domain, it is necessary to identify additional structure to obtain positive results. We mostly consider structural properties of the neighborhoods of the states of nature.

**Unique Accepts and Rejects.** In an instance with *unique accepts*, there is a single acceptable state, i.e.,  $|\Theta_A| = 1$ . Similarly, for *unique rejects* we have  $|\Theta_R| = 1$ . This is equivalent to assuming that every acceptable (rejectable, resp.) state  $\theta$  has the same neighborhood  $N(\theta)$ .

**Degree-bounded States.** In an instance with *degree- $k$  states*, every state  $\theta \in \Theta$  has  $|N(\theta)| \leq k$ . Similarly, for *degree- $k$  accepts*, every acceptable state  $\theta \in \Theta_A$  has  $|N(\theta)| \leq k$ , and for *degree- $k$  rejects* every rejectable state  $\theta \in \Theta_R$  has  $|N(\theta)| \leq k$ .

**Foresight.** Sher [25] considers instances with *foresight* defined as follows. For an acceptable state  $\theta \in \Theta_A$ , a signal  $\sigma \in N(\theta)$  is called *minimally forgeable for  $\theta$*  if  $\sigma \in N(\theta')$  implies  $\sigma' \in N(\theta')$  for every other signal  $\sigma' \in N(\theta)$  and every rejectable state  $\theta' \in \Theta_R$ . In an instance *with foresight* every acceptable state has a minimally forgeable signal. Intuitively, in such a problem every acceptable state  $\theta$  has a (not necessarily unique) signal that is maximally informative about  $\theta$  with respect to the set of rejectable states. Foresight strictly generalizes other properties studied in previous work, e.g. normality [4]. Normality requires a signal for every state (not only the acceptable ones) that satisfies the condition of minimally forgeable, and it satisfies the condition w.r.t. all states (not only w.r.t. rejectable ones). In addition, foresight is a generalization of instances with unique rejects, as well as a generalization the class of degree-1 accepts.

■ **Table 1** Approximation results shown in this paper, as well as results shown or implied by [25].

Scenario	Constrained Delegation		Constrained Persuasion	
	Upper	Lower	Upper	Lower
General	2	$2 - \varepsilon$ ( $P \neq NP$ )	$O(n)$	$n^{1-\varepsilon}$ ( $P \neq NP$ )
Degree-2 States	1.1	APX-hard [25]	$O(n)$	$n^{1-\varepsilon}$ ( $P \neq NP$ )
Degree- $d$ States	$2 - 1/d^2$	APX-hard [25]	$O(n)$	$n^{1-\varepsilon}$ ( $P \neq NP$ )
Degree-1 Rejects	2	APX-hard [25]	$O(n)$	$n^{1-\varepsilon}$ ( $P \neq NP$ )
Degree-1 Accepts	1 [25]		$O(n)$	$n^{1-\varepsilon}$ ( $P \neq NP$ )
Foresight	1 [25]		$O(n)$	$n^{1-\varepsilon}$ ( $P \neq NP$ )
Unique Rejects	1 [25]		1	
Unique Accepts	1		PTAS	Strongly NP-hard

## 2.2 Results and Contribution

We provide polynomial-time exact and approximation algorithms as well as hardness results for the general problems and the domains with more structure described above.

We first consider the case of the constrained equilibrium problem. The existence of a sequential equilibrium is implied by [14]; we show that it can always be computed in polynomial time by repeatedly solving a maximum flow problem. We compare the utility obtained in an equilibrium with the one achievable with commitment power, for the sender and the receiver, respectively. Formally, we define and bound the ratio of the utilities for best and worst-case equilibria, in the spirit of prices of anarchy and stability. For the receiver, it is known that the price of stability is 1 [14]; we show that the price of anarchy is 2. For the sender we show that both ratios are unbounded. This substantial utility gain provides further motivation to study problems with commitment power.

Our results for constrained delegation and persuasion are summarized in Table 1. We discuss a selected subset of our most interesting contributions in the main part of the paper. All missing proofs are deferred to the full version of this paper. In addition, in the full version, we prove additional results that omitted from this version due to spatial constraints.

For the constrained delegation problem, we show two interesting non-trivial approximation results. For degree-2 states, we propose a semidefinite-programming algorithm to compute a 1.1-approximation. To the best of our knowledge, this is the first application of advanced results from the SDP toolbox in the context of information design, as well as mechanism design. For instances with degree- $d$  states we give a  $(2 - \frac{1}{d^2})$ -approximation algorithm via LP rounding.

For constrained persuasion, the strong hardness arises from deciding which action should be preferred by the receiver for each signal. It holds even in several seemingly special cases with degree-1 accepts, degree-1 rejects and degree-2 states. As a consequence, good approximation algorithms can be obtained only in significantly more limited scenarios than for delegation. For unique accepts, we prove strong NP-hardness (i.e. there is no FPTAS unless  $P = NP$ ) and provide a polynomial-time approximation scheme (PTAS).

## 2.3 Related Work

There is a large literature on strategic communication, see Sobel [27] for an extensive review. The works most closely related to ours are [14, 25]. Glazer and Rubinstein [14] introduce the problem of constrained delegation. They show, among other things, that the optimal decision scheme in constrained delegation is deterministic. Furthermore, they prove that there is

always a sequential equilibrium where the receiver plays the optimal decision scheme from constrained delegation, i.e., the price of stability for the receiver is 1. This condition is termed “credibility”. It is easy to see that this is not true when sender moves first. This conceptual difference between persuasion and delegation is reflected as a difference in the problems’ computational complexity. Deterministic optimal strategies and “credibility” hold also beyond the simple model with 2 actions – when receiver utility is a concave transformation of sender utility, see [24]. Sher [25] builds on the model of [14] and characterizes optimal rules for static as well as dynamic persuasion. Furthermore, and more relevant to our interest here, he proves an NP-hardness result for constrained delegation, as well as provides a polynomial-time algorithm for optimal delegation in instances with foresight. Here we strengthen this hardness result to a hardness of approximation within a factor of  $2 - \varepsilon$  (and provide a matching, alas trivial, approximation algorithm). While this subsumes NP-hardness in general, we observe that his hardness proof applies in case of degree-2 states and degree-1 rejects, and that it even implies APX-hardness for such instances.

Glazer and Rubinstein [13] study a related setting, where the state of nature is multi-dimensional, and the receiver can verify at most one dimension. The authors characterize the optimal mechanism as a solution to a particular linear programming problem, show that it takes a fairly simple form, and show that random mechanisms may be necessary to achieve the optimum. Carroll and Egorov [5] study the problem of fully revealing the sender’s information in a setting with multidimensional states, where the receiver can verify a single dimension. Importantly, the dimension the receiver chooses to reveal depends on the sender’s message.

A number of works in the algorithmic economics literature investigate the computational complexity of persuasion and information design. Computational aspects of the Bayesian persuasion model [19] are studied in, e.g., [10, 6, 9, 8, 11, 18, 17], but in these works there are no limits on the senders’ signals, i.e.,  $H$  is the complete bipartite graph. More closely related to our work are [7, 16] who study computational problems in Bayesian persuasion with limited signals, where the number of signals is smaller than the number of actions.

### 3 Sequential Equilibria

We first study the scenario without commitment power. Our interest here is to obtain a signaling scheme  $\varphi : E \rightarrow [0, 1]$  and a decision scheme  $\psi : \Sigma \rightarrow [0, 1]$ , such that the pair  $(\varphi, \psi)$  forms a sequential equilibrium.

► **Theorem 4.** *A sequential equilibrium can be computed in polynomial time.*

Our algorithm repeatedly sets up a flow network based on the graph  $H$ . In each iteration, it computes a maximum  $s$ - $t$  flow and identifies suitable regions of the graph where it fixes the equilibrium schemes of sender and receiver. Then it removes the fixed regions and repeats the construction on the graph with the remaining states and signals. After at most  $\min\{n, m\}$  iterations, the algorithm finishes the construction of the equilibrium.

How desirable is an equilibrium for the sender and the receiver? By how much can each player benefit when he or she enjoys commitment power? Towards this end, we bound the ratios of the optimal utility achievable with commitment power over the utilities in the worst and best equilibrium. Intuitively, commitment power might be interpreted as a form of control over the game, so we use the term *price of anarchy* and *price of stability* to refer to the ratios, respectively.

More formally, for the price of anarchy we bound the ratio of the optimal utility achievable with commitment over the *worst* utility in any sequential equilibrium. For the price of stability we bound the ratio of the optimal utility achievable with commitment over the *best* utility in any sequential equilibrium.

For the receiver, the optimal scheme with commitment leads to an equilibrium [14], so the price of stability is 1. The price of anarchy is 2 (c.f. Proposition 7 below). For the sender, both prices of anarchy and stability are unbounded.

► **Proposition 5.** *The price of anarchy for the receiver is 2 and this is tight. The prices of anarchy and stability for the sender are unbounded.*

## 4 Constrained Delegation

In constrained delegation, the game starts with the receiver committing to a decision scheme  $\psi : \Sigma \rightarrow [0, 1]$ , where  $\psi(\sigma)$  is the probability to choose action A if the sender reports signal  $\sigma$ . The first insight is due to [14, Proposition 1].

► **Lemma 6** (Glazer and Rubinstein [14]). *In constrained delegation, there is an optimal decision scheme  $\psi^*$  that is deterministic, i.e.,  $\psi^*(\sigma) \in \{0, 1\}$  for all  $\sigma \in \Sigma$ .*

Given a deterministic decision scheme  $\psi$ , the sender's problem is trivial: after learning  $\theta$ , report an arbitrary signal  $\sigma \in N(\theta)$  such that  $\psi(\sigma) = 1$  if one exists. Otherwise, report an arbitrary signal  $\sigma \in N(\theta)$ . In the following, we focus on the computational complexity of the receiver's problem: How hard is it to compute the optimal  $\psi$ ? What about a good approximation algorithm?

This problem turns out to be much easier than the sender's problem in constrained persuasion studied below. It readily admits a trivial 2-approximation algorithm. Let  $\psi_A$  be the scheme that accepts all signals, i.e.,  $\psi_A(\sigma) = 1$  for all  $\sigma$ , and  $\psi_R$  the scheme that rejects all signals. The better of  $\psi_A$  and  $\psi_R$  results in utility  $\max\{q_A, q_R\}$  for the receiver, which is at least  $1/2$ . Clearly, the receiver can obtain at most a utility of 1.

► **Proposition 7.** *For constrained delegation, the better of  $\psi_A$  and  $\psi_R$  is a 2-approximation to the optimal decision scheme  $\psi^*$ .*

In Section 4.1 we show that the factor 2 is essentially optimal in the worst case, unless  $P = NP$ . In Section 4.2 we present our results on approximation algorithms.

### 4.1 Hardness

Sher [25, Theorem 7] shows NP-hardness of constrained delegation, even in the special case with degree-1 rejects and degree-2 states. His construction can be extended easily to show APX-hardness (we provide the details in the full version of this paper). Our main result in this section is a stronger hardness result that matches the guarantee of the trivial algorithm in Proposition 7.

► **Theorem 8.** *For any constant  $\varepsilon \in (0, 1)$ , it is NP-hard to approximate constrained delegation within a factor of  $(2 - \varepsilon)$ .*

For simplicity, we sketch below an outline for a reduction that does *not* give the NP-hardness, but nonetheless encapsulates the main ideas of the proof. After the outline, we roughly explain the changes needed to achieve the NP-hardness; the full proof is deferred to the full version of this paper.



We reduce from the BIPARTITE VERTEX EXPANSION problem. In this problem, we are given a bipartite graph  $(U, V, E)$  and positive real number  $\beta$ . The goal is to select (at least)  $\beta|U|$  vertices from  $U$  such that their neighborhood (in  $V$ ) is as small as possible. Khot and Saket [20] show the following strong inapproximability result:

► **Theorem 9** ([20]). *Assuming  $\text{NP} \not\subseteq \bigcap_{\delta>0} \text{DTIME}(2^{n^\delta})$ , for any positive constants  $\tau, \gamma > 0$ , there exists  $\beta \in (0, 1)$  such that no polynomial-time algorithm can, given a bipartite graph  $(U, V, E)$ , distinguish between the following two cases:*

- (YES) *There exists  $S^* \subseteq U$  of size at least  $\beta|U|$  where  $|N(S^*)| \leq \gamma|V|$ .*
- (NO) *For any  $S \subseteq U$  of size at least  $\tau\beta|U|$ ,  $|N(S)| > (1 - \gamma)|V|$ .*

The main idea of our reduction is as follows. Roughly speaking, given a bipartite graph  $(U, V, E)$ , we set  $\Sigma = U$ ,  $\Theta_R = V$  and the edge set between them is exactly  $E$ . To get a high utility on  $\Theta_R$ , we must pick a signal set  $T \subseteq \Sigma$  such that  $|N(T)|$  is small, and set  $\psi(\sigma) = 1$  for all  $\sigma \in T$ ; this does not mean much so far, since we could just pick  $T = \emptyset$ . This is where the set of acceptable states comes in: we let  $\Theta_A$  be equal to  $U^\ell = \{(u_1, \dots, u_\ell) \mid u_i \in U\}$  for some appropriate  $\ell \in \mathbb{N}$ , and there is an edge between  $\theta = (u_1, \dots, u_\ell)$  and  $\sigma = u$  if  $u_i = u$  for some  $i \in [\ell]$ . Intuitively, this forces us to pick  $T$  that is not too small as otherwise  $\Theta_A$  won't contribute to the total utility. Finally, we need to pick a distribution  $\mathcal{D}$  over  $\Theta$  such that  $q_A = q_R$ , as otherwise the trivial algorithm already gets better than a 2-approximation.

As stated earlier, the above reduction does not yet give NP-hardness, because Theorem 9 relies on a stronger assumption<sup>1</sup> that  $\text{NP} \not\subseteq \bigcap_{\delta>0} \text{DTIME}(2^{n^\delta})$ . To overcome this, we instead use a “colored version” of the problem, where every vertex in  $U$  is colored and the subset  $S \subseteq U$  must only contain vertices of different colors (i.e., be “colorful”). It turns out that the above reduction can be adapted to work with such a variant as well, by changing the acceptable states  $\Theta_A$  to “test” this condition instead of the condition that  $|S|$  is small. Furthermore, we show, via a reduction from the Label Cover problem, that this colored version of BIPARTITE VERTEX EXPANSION is NP-hard to approximate. Together, these imply Theorem 8. Our proof formalizes this outline; see the full version for details.

## 4.2 Approximation Algorithms for Constrained Delegation

By Theorem 8 there is no hope for a  $(2 - \epsilon)$ -approximation algorithm for the constrained delegation problem. Proposition 7 provides a matching guarantee.

As a consequence, we examine in which way instance parameters influence the existence of polynomial-time approximation algorithms. In particular, the maximum degree  $d$  is a main force that drives the hardness result. For the case of degree at most  $d$ , we give a  $2 - \frac{1}{d^2}$  approximation algorithm via LP rounding. When  $d = 2$ , we improve upon this by giving a 1.1-approximation algorithm via SDP rounding.

### 4.2.1 Better than 2 via LP Rounding

For instances with degree- $d$ -states we take the better of (1) rounding the natural linear program for constrained delegation and (2) the trivial scheme of Proposition 7.

► **Theorem 10.** *For constrained delegation with degree- $d$  states there is a polynomial-time  $(2 - \frac{1}{d^2})$ -approximation algorithm.*

<sup>1</sup> We remark that it is entirely possible that Theorem 9 holds under NP-hardness (instead of under the assumption  $\text{NP} \not\subseteq \bigcap_{\delta>0} \text{DTIME}(2^{n^\delta})$ ) but this is not yet known.



**Proof.** Consider the following integer program for constrained delegation (c.f. [14, 25]).

$$\max \sum_{\theta \in \Theta} c_{\theta} q_{\theta} \quad (1a)$$

$$\text{s.t.} \quad \sum_{\sigma \in N(\theta)} \psi_{\sigma} \geq c_{\theta}, \quad \text{for all } \theta \in \Theta_A \quad (1b)$$

$$\sum_{\sigma \in N(\theta)} \psi_{\sigma} \leq |N(\theta)|(1 - c_{\theta}) \quad \text{for all } \theta \in \Theta_R \quad (1c)$$

$$\psi_{\sigma} \in \{0, 1\}, \text{ for all } \sigma \in \Sigma \quad \text{and} \quad c_{\theta} \in \{0, 1\}, \text{ for all } \theta \in \Theta \quad (1d)$$

The variable  $\psi_{\sigma}$  encodes whether the action is accept or reject for signal  $\sigma$ . The variable  $c_{\theta}$  encodes whether the receiver makes the correct choice when the state of nature is  $\theta$ . Constraint (1b) states that, if  $\theta \in \Theta_A$ , she can't make the correct choice when she rejects all signals available from  $\theta$ . Constraint (1c) states that, if  $\theta \in \Theta_R$ , making the correct choice means rejecting all signals available from  $\theta$ ; the  $|N(\theta)|$  term ensures that the constraint can still be satisfied even when  $c_{\theta} = 0$ .

Our algorithm first solves the linear relaxation of this integer program; let  $\hat{\psi}_{\sigma}$  and  $\hat{c}_{\theta}$  be the fractional optimum. We round this solution by setting  $\psi_{\sigma} = 1$  with probability  $\hat{\psi}_{\sigma}$ , and 0 otherwise. We can optimally pick  $c_{\theta}$  given the  $\psi_{\sigma}$ 's. The rounded solution is feasible by definition; we show that it is a good approximation to the optimal LP value, i.e.,  $\sum_{\theta \in \Theta} \hat{c}_{\theta} q_{\theta}$ .

Let  $G = \frac{1}{|\Theta_A|} \sum_{\theta \in \Theta_A} \hat{c}_{\theta} q_{\theta}$  and  $B = \frac{1}{|\Theta_R|} \sum_{\theta \in \Theta_R} \hat{c}_{\theta} q_{\theta}$  be the average contribution to the LP objective from the acceptable and rejectable states, respectively. The LP value is  $G|\Theta_A| + B|\Theta_R|$ . We start by showing the following lower bound on the expected value of the rounded solution.

► **Lemma 11.**  $\mathbb{E}[\sum_{\theta \in \Theta} c_{\theta} q_{\theta}] \geq \frac{G|\Theta_A|}{d} + q_R(1 - d) + dB|\Theta_R|$ .

**Proof.** First, consider a state  $\theta \in \Theta_A$ . The probability that  $c_{\theta} = 1$  is at least the probability that we rounded one of the  $\psi_{\sigma}$  variables to 1, for  $\sigma \in N(\theta)$ , i.e.,

$$\Pr[c_{\theta} = 1] \geq \max_{\sigma \in N(\theta)} \hat{\psi}_{\sigma} \geq \frac{\hat{c}_{\theta}}{|N(\theta)|} \geq \frac{\hat{c}_{\theta}}{d}, \quad (2)$$

where we used the fact that  $\hat{c}_{\theta}$  satisfies Constraint (1b). For a state  $\theta \in \Theta_R$ , the probability that  $c_{\theta} = 1$  is exactly the probability that none of its signals were selected, which is  $\prod_{\sigma \in N(\theta)} (1 - \hat{\psi}_{\sigma}) \geq 1 - \sum_{\sigma \in N(\theta)} \hat{\psi}_{\sigma}$ . Thus

$$\Pr[c_{\theta} = 1] \geq 1 - \sum_{\sigma \in N(\theta)} \hat{\psi}_{\sigma} \geq 1 - |N(\theta)|(1 - \hat{c}_{\theta}) \geq 1 - d + d\hat{c}_{\theta}, \quad (3)$$

where we used the fact that  $\hat{c}_{\theta}$  satisfies Constraint (1c). Adding up (2) and (3), the expected value of our rounded solution is

$$\mathbb{E}\left[\sum_{\theta \in \Theta} c_{\theta} q_{\theta}\right] \geq \sum_{\theta \in \Theta_A} \frac{q_{\theta} \hat{c}_{\theta}}{d} + \sum_{\theta \in \Theta_R} q_{\theta} (1 - d + d\hat{c}_{\theta}) \geq \frac{G|\Theta_A|}{d} + q_R(1 - d) + dB|\Theta_R|. \quad \blacktriangleleft$$

Our final algorithm, i.e., the better of the trivial scheme and the rounded LP solution, has expected value at least  $\max\{q_A, q_R, \mathbb{E}[\sum_{\theta \in \Theta} c_\theta q_\theta]\}$ . We have that

$$\begin{aligned} \left(2d - \frac{1}{d}\right) \max \left\{ q_A, q_R, \mathbb{E} \left[ \sum_{\theta \in \Theta} c_\theta q_\theta \right] \right\} &\geq \left(d - \frac{1}{d}\right) q_A + (d-1)q_R + \mathbb{E} \left[ \sum_{\theta \in \Theta} c_\theta q_\theta \right] \\ &\stackrel{\text{Lemma 11}}{\geq} \left(d - \frac{1}{d}\right) q_A + (d-1)q_R + \frac{G|\Theta_A|}{d} + q_R(1-d) + dB|\Theta_R| \\ &\stackrel{(G|\Theta_A| \leq q_A)}{\geq} dG|\Theta_A| + dB|\Theta_R|, \end{aligned}$$

which is  $d$  times the value of the optimum fractional value of the LP. The theorem follows.  $\blacktriangleleft$

#### 4.2.2 Better than 2 via Semidefinite Programming

In this subsection we give a 1.1-approximation algorithm for constrained delegation with degree-2 states, where every state of nature  $\theta$  has at most two allowed signals,  $\sigma_u$  and  $\sigma_v$ . The approach stems from an observation that the problem belongs to the class of *constraint satisfaction problems* (CSPs); we make use of the toolbox for semidefinite program (SDP) rounding in approximating CSPs (e.g. [15, 12, 21]).

Consider the integer program (4a) for our problem below. We assume w.l.o.g. that every state has *exactly* two adjacent signals; if there is a state  $\theta$  with a single neighbor  $\sigma$ , we can add a parallel edge  $(\theta, \sigma)$  in  $H$  and the analysis remains valid. Note that the integer program here is *not* the same as the one used in the previous subsection. An intuitive reason for the change is that the variables  $c_\theta$  there are redundant: given  $\{\psi_\sigma\}_{\sigma \in \Sigma}$ , the values of  $\{c_\theta\}_{\theta \in \Theta}$  are already fixed. In particular, each  $c_\theta$  can be expressed as a degree- $d$  polynomial<sup>2</sup> in  $\{\psi_\sigma\}_{\sigma \in N(\theta)}$ , which is exactly how the integer program below is written.

$$\max_{x \in \{-1, 1\}^m} \quad \frac{1}{4} \sum_{\substack{\theta \in \Theta_A \\ \theta = (\sigma_i, \sigma_j)}} (3 - x_i - x_j - x_i x_j) q_\theta + \frac{1}{4} \sum_{\substack{\theta \in \Theta_R \\ \theta = (\sigma_i, \sigma_j)}} (1 + x_i + x_j + x_i x_j) q_\theta \quad (4a)$$

In the program above  $x_i = -1$  is interpreted as accepting when the signal is  $\sigma_i$ . One can check that  $\frac{1}{4}(3 - x_i - x_j - x_i x_j)$  is equal to 1 iff at least one of  $x_i, x_j$  is  $-1$  (and zero otherwise), i.e., a state of nature  $\theta \in \Theta_A$  contributes to the objective only when at least one of its allowed signals is accepted. Similarly,  $\frac{1}{4}(1 + x_i + x_j + x_i x_j)$  is equal to 1 if and only if both  $x_i$  and  $x_j$  are equal to 1.

We will solve the semidefinite relaxation of this program, and give a rounding algorithm. The SDP is the following, where we replaced  $x_i$  by  $w_i$ , to distinguish these vector variables from the variables of our integer program above.

<sup>2</sup> Note that linear functions do not suffice to express  $c_\theta$ . In particular, if we rewrite (1c) for  $\theta = (\sigma_i, \sigma_j)$  as  $c_\theta \leq 1 - \frac{\psi_{\sigma_i} + \psi_{\sigma_j}}{2}$ , then it is still possible to have  $c_\theta = 1/2$  when  $\psi_{\sigma_i} = 1, \psi_{\sigma_j} = 0$ .

$$\begin{aligned} \max \quad & \frac{1}{4} \sum_{\theta=(\sigma_i, \sigma_j) \in \Theta_A} (3 - w_i \cdot w_0 - w_j \cdot w_0 - w_i \cdot w_j) q_\theta \\ & + \frac{1}{4} \sum_{\theta=(\sigma_i, \sigma_j) \in \Theta_R} (1 + w_i \cdot w_0 + w_j \cdot w_0 + w_i \cdot w_j) q_\theta \end{aligned} \quad (5a)$$

$$\text{s.t.} \quad w_i \cdot w_i = 1 \quad \text{for all } i \in [m] \cup \{0\} \quad (5b)$$

$$w_i \cdot w_0 + w_j \cdot w_0 + w_i \cdot w_j \geq -1 \quad \text{for all } i, j \in [m] \quad (5c)$$

$$-w_i \cdot w_0 + w_j \cdot w_0 - w_i \cdot w_j \geq -1 \quad \text{for all } i, j \in [m] \quad (5d)$$

$$-w_i \cdot w_0 - w_j \cdot w_0 + w_i \cdot w_j \geq -1 \quad \text{for all } i, j \in [m] \quad (5e)$$

$$w_i \in \mathbb{R}^{m+1} \quad \text{for all } i \in [m] \cup \{0\}$$

Constraint (5b) is standard. Constraints (5c)-(5e) encode the triangle inequalities, which are satisfied by every valid solution to the original program; these strengthen the relaxation a bit (see [12, 21]). Let  $\mathcal{V}_{SDP}$  denote the optimal value of this semidefinite program (SDP). We generally cannot find the exact solution to an SDP, but it is possible to find a feasible solution with value at least  $\mathcal{V}_{SDP} - \epsilon$  in time polynomial in  $1/\epsilon$  (see [1]). In our analysis we will (as is typically the case) ignore the  $\epsilon$  factor as it can be made arbitrarily small given sufficient time.

It is known that the SDP written above provides the optimal approximation achievable in polynomial time for any 2-CSPs [22, 23] including our problem, assuming the Unique Games Conjecture (UGC). However, a generic rounding algorithm from this line of work (see e.g. [23]) does not give a concrete approximation ratio. Below, we describe a specific family of rounding algorithms for which we can provide the concrete approximation ratio of 1.1.

## Rounding Algorithm

Given solution vectors  $\{w_0, w_1, \dots, w_m\}$ ,  $w_i \in \mathbb{R}^{m+1}$ , for this SDP we produce a feasible solution  $x_i \in \{-1, 1\}$  (for  $i \in [m]$ ) to the original integer program as follows. Let  $\xi_i = w_0 \cdot w_i$ , and  $\tilde{w}_i = \frac{w_i - \xi_i w_0}{\sqrt{1 - \xi_i^2}}$  be the part of  $w_i$  orthogonal to  $w_0$ , normalized to a unit vector. Our rounding algorithm mostly follows the rounding procedure of [21], which they call  $\mathcal{THRESH}^-$ . First, pick a  $(m+1)$ -dimensional vector<sup>3</sup>  $r \sim \mathcal{N}(0, 1)$   $r \in \mathbb{R}^{m+1}$ . Then, set  $x_i = -1$  (which corresponds to accepting signal  $\sigma_i$ ) if and only if  $\tilde{w}_i \cdot r \leq T(\xi_i)$ , where  $T(\cdot)$  is a threshold function, and set  $x_i = 1$  otherwise. Specifically,  $T(x) = \Phi^{-1}(\frac{1-\nu(x)}{2})$ , where  $\Phi^{-1}(\cdot)$  is the inverse of the normal distribution function, and  $\nu : [-1, 1] \rightarrow [-1, 1]$  is a function. Later in the analysis – and this is essentially the point in which various SDP rounding methods diverge from each other, e.g. see [26] for the different choices for MAX-2-SAT and MAX-2-AND – we will optimize over a family of  $\nu(\cdot)$ , exploiting structure in our problem, in order to improve our approximation ratio.

## Generic Analysis

We now derive a generic analysis for  $\mathcal{THRESH}^-$  algorithms; note that these are similar arguments as in [21, 2]. However, in the end, we will pick a different function  $\nu$  than previous works, which results in better approximation ratios for our problem.

<sup>3</sup> In other words, the  $i$ -th dimension  $r_i$  is sampled independently from a Gaussian with zero mean and variance one.

First, notice that  $\tilde{w}_i \cdot r$  is a standard  $\mathcal{N}(0, 1)$  variable, and therefore by the choice of  $T(\cdot)$  we have that  $\Pr[x_i = -1] = \frac{1 - \nu(\xi_i)}{2}$ , which implies that

$$\mathbb{E}[x_i] = \nu(\xi_i) . \quad (6)$$

Now, we need to also analyze the quadratic terms. Let  $\Gamma_c(\mu_1, \mu_2) = \Pr[X_1 \leq t_1 \text{ and } X_2 \leq t_2]$ , where  $t_i = \Phi^{-1}(\frac{1 - \mu_i}{2})$ , and  $X_1, X_2 \in \mathcal{N}(0, 1)$  with covariance  $c$  (in other words,  $\Gamma_c$  is the bivariate normal distribution function with covariance  $c$ , with a transformation on the input).

Let  $\rho = w_i w_j$  and  $\tilde{\rho} = \tilde{w}_i \tilde{w}_j = \frac{\rho - \xi_i \xi_j}{\sqrt{1 - \xi_i^2} \sqrt{1 - \xi_j^2}}$ . Observe that the products  $\tilde{w}_i \cdot r$  and  $\tilde{w}_j \cdot r$  are  $\mathcal{N}(0, 1)$  random variables with covariance  $\tilde{\rho}$ . Thus, the probability that  $\tilde{w}_i \cdot r \leq T(\xi_i)$  and  $\tilde{w}_j \cdot r \leq T(\xi_j)$  (i.e., both  $x_i, x_j$  are set to  $-1$ ) is exactly  $\Gamma_{\tilde{\rho}}(\nu(\xi_i), \nu(\xi_j))$ . The probability that  $x_i = x_j = 1$  is equal to  $\Gamma_{\tilde{\rho}}(-\nu(\xi_i), -\nu(\xi_j))$ . Austrin [2, Proposition 2.1] shows that  $\Gamma_c(-\mu_1, -\mu_2) = \Gamma_c(\mu_1, \mu_2) + \mu_1/2 + \mu_2/2$ . Using this fact we can calculate the probability that  $x_i = x_j$ , which, in turn, gives that

$$\mathbb{E}[x_i x_j] = 4\Gamma_{\tilde{\rho}}(\nu(\xi_i), \nu(\xi_j)) + \nu(\xi_i) + \nu(\xi_j) - 1 . \quad (7)$$

With Equations (6) and (7) at hand we can calculate the expected value of our rounding algorithm (i.e., the expected value of (4a)) for every choice of  $\nu$ , and compare it against the value of the SDP in (5a). Specifically, we will aim for a term-by-term approximation. Define the following quantities:

$$\begin{aligned} \ell_\nu^{OR}(\xi_i, \xi_j, \rho) &= \frac{3 - \xi_i - \xi_j - \rho}{4 - 2\nu(\xi_i) - 2\nu(\xi_j) - 4\Gamma_{\tilde{\rho}}(\nu(\xi_i), \nu(\xi_j))} \\ \ell_\nu^{AND}(\xi_i, \xi_j, \rho) &= \frac{1 + \xi_i + \xi_j + \rho}{2\nu(\xi_i) + 2\nu(\xi_j) + 4\Gamma_{\tilde{\rho}}(\nu(\xi_i), \nu(\xi_j))} , \end{aligned}$$

and let

$$\ell^{OR}(\nu) = \min_{\xi_i, \xi_j, \rho} \ell_\nu^{OR}(\xi_i, \xi_j, \rho) \quad \text{and} \quad \ell^{AND}(\nu) = \min_{\xi_i, \xi_j, \rho} \ell_\nu^{AND}(\xi_i, \xi_j, \rho) ,$$

where the minimization is over all choices of  $\xi_i, \xi_j, \rho \in [-1, 1]$  that satisfy the triangle inequalities (Constraints (5c)-(5e)). It is now straightforward to see that the term-by-term analysis implies that, for any choice of  $\nu$ , our approximation ratio is at most  $\max\{\ell^{OR}(\nu), \ell^{AND}(\nu)\}$ .

### Choosing $\nu$ and Putting Things Together

We are left to choose the function  $\nu$  that results in the smallest approximation ratio  $\max\{\ell^{OR}(\nu), \ell^{AND}(\nu)\}$ . We consider a rounding function of the form  $\nu(y) = \alpha \cdot y + \beta$  for parameters  $\alpha, \beta$  to be chosen. Using extensive computational effort, we found that  $\alpha = 0.8825$  and  $\beta = 0.0384$  perform well. Once we have a choice for  $\alpha$  and  $\beta$ , it remains to prove the approximation ratio.

We have a computer-assisted proof showing that the approximation ratio is at most 1.1; our computer-based proof approach is similar to that of [26]. Roughly speaking, we divide the cube  $(\xi_i, \xi_j, \rho) \in [-1, 1]^3$  into a certain number of subcubes. For each subcube, we (numerically) compute an upper bound to  $\max\{\ell_\nu^{OR}(\xi_i, \xi_j, \rho), \ell_\nu^{AND}(\xi_i, \xi_j, \rho)\}$ . If this upper bound is already at most 1.1, then we are finished with the subcube. Otherwise, we divide it further into a certain number of subcubes. By continuing this process, we eventually manage to show that for the whole region  $[-1, 1]^3$  that satisfies the triangle inequalities, the ratio must be at most 1.1, as desired. (The smallest subcube our proof considers has edge length 0.00078.)

## Comparison to Prior Work

As stated earlier, our algorithm, with the exception of the choice of  $\nu$ , is similar to [21] and the follow-up works (e.g. [2, 26]). However, perhaps surprisingly, we end up with a better approximation ratio than the MAX 2-AND problem<sup>4</sup>, whose approximation ratio is known to be at least 1.143 assuming the UGC [3]. To understand the difference, recall that MAX 2-AND can be written as  $\max \frac{1}{4} \sum_{(i,j,b_i,b_j)} (1 + b_i x_i + b_j x_j + b_i b_j x_i x_j)$  where  $b_i, b_j \in \{\pm 1\}$  (representing whether the variable is negated in the clause). This is very similar to our problem (4a), except that MAX 2-AND has the aforementioned  $b_i, b_j$ -terms for negation. It turns out that this is also the cause that we can achieve better approximation ratio. Specifically, these negation terms led previous works [21, 2, 26, 3] to only consider  $\nu$  that is an odd function, i.e.,  $\nu(y) = \nu(-y)$  for all  $x \in [-1, 1]$ . For example, Austrin [2] considers a function of the form  $\nu(y) = \alpha \cdot y$ . We note here that, due to the aforementioned UGC-hardness of MAX 2-AND, we cannot hope to get an approximation ratio smaller than 1.143 using odd  $\nu$ . Nonetheless, since we do not have “negation” in our problem, we are not only restricted to odd  $\nu$ , allowing us to consider a more general family of the form  $\nu(y) = \alpha \cdot y + \beta$  for  $\beta \neq 0$ . This ultimately leads to our better approximation ratio.

## 5 Constrained Persuasion

Let us now turn to the constrained persuasion problem. The sender first commits to a signaling scheme  $\varphi$ , which she then uses to transmit information to the receiver, once the state of nature is revealed. Given that the sender has commitment power and the receiver knows  $\varphi$ , the receiver picks action A if and only if conditioned on receiving signal  $\sigma$ , the expected utility of A is more than R, i.e.,

$$\sum_{\theta \in N(\sigma) \cap \Theta_A} \varphi(\theta, \sigma) \geq \sum_{\theta \in N(\sigma) \cap \Theta_R} \varphi(\theta, \sigma)$$

or, equivalently,  $2 \cdot \sum_{\theta \in N(\sigma) \cap \Theta_A} \varphi(\theta, \sigma) \geq \sum_{\theta \in N(\sigma)} \varphi(\theta, \sigma)$ .

In this case, we say that  $\sigma$  is an *accept signal*, otherwise we call  $\sigma$  a *reject signal*. An optimal signaling scheme  $\varphi^*$  maximizes the expected utility of the sender, i.e., the total probability associated with accept signals. Note that if both accepting and rejecting are optimal actions for the receiver, we assume that she breaks ties in favor of the sender (so, in our case, accept). This mild assumption is standard in economic bilevel problems (e.g., when indifferent between buying and not buying, a potential customer is usually assumed to buy) and is often without loss of generality. This way we avoid obfuscating technicalities in the definition of optimal schemes  $\varphi^*$ .

We study the computational complexity of finding  $\varphi^*$  and polynomial-time approximation algorithms. In general, approximating  $\varphi^*$  can be an extremely hard problem, even in the constrained persuasion problem. Our first insight in Section 5.1 is that the main source of hardness in the problem is deciding the optimal set of accept signals. We then provide a simple  $2n$ -approximation algorithm and a  $n^{1-\epsilon}$ -hardness in Section 5.2. The PTAS and the matching strong NP-hardness for instances with unique accepts is discussed in Section 5.3.

<sup>4</sup> This is the problem where we are given a set of clauses, each of which is an AND of two literals. The goal is to assign the variables as to maximize the number of satisfied clauses.

### 5.1 Signal Partitions

A signaling scheme  $\varphi$  partitions the signal space  $\Sigma$  into  $(\Sigma_A, \Sigma_R)$ , in the sense that the receiver takes action  $A$  if and only if she gets signal  $\sigma \in \Sigma_A$  (and  $R$  for  $\Sigma_R$ ). Determining this partition of the signal set turns out to be the main source of computational hardness of finding  $\varphi^*$ : Given an optimal partition of the signal set, the reduced problem of computing appropriate optimal signaling probabilities is solved with a linear program.

We prove this result in a general case of the persuasion problem, in which the receiver has an arbitrary finite set  $\mathcal{A}$  of actions. Moreover, sender and receiver can have utilities  $u_s, u_r : \mathcal{A} \times \Theta \rightarrow \mathbb{R}$  that yield arbitrary positive or negative values for every (state of nature, action)-pair.

► **Proposition 12.** *Given a partition  $P = (\Sigma_a)_{a \in \mathcal{A}}$  of the signal space such that the receiver's best action for a signal  $\sigma \in \Sigma_a$  is action  $a$ , an optimal signaling scheme  $\varphi_P^*$  for the general persuasion problem that (1) implements these receiver preferences and (2) maximizes the sender utility, can be computed by solving a linear program of polynomial size.*

**Proof.** Given  $P = (\Sigma_a)_{a \in \mathcal{A}}$ , consider the following linear program (8).

$$\begin{aligned}
 \text{Max.} \quad & \sum_{a \in \mathcal{A}} \sum_{\sigma \in \Sigma_a} \sum_{\theta \in N(\sigma)} x_{\theta, \sigma} \cdot u_s(a, \theta) \\
 \text{s.t.} \quad & \sum_{\theta \in N(\sigma)} x_{\theta, \sigma} \cdot u_r(a, \theta) \geq \sum_{\theta \in N(\sigma)} x_{\theta, \sigma} \cdot u_r(a', \theta) \quad \text{for all } a \in \mathcal{A}, \sigma \in \Sigma_a, a' \in \mathcal{A} \\
 & \sum_{\sigma \in N(\theta)} x_{\theta, \sigma} = q_\theta \quad \text{for all } \theta \in \Theta \\
 & x_{\theta, \sigma} \geq 0 \quad \text{for all } \sigma \in \Sigma, \theta \in N(\sigma)
 \end{aligned} \tag{8}$$

For each  $\sigma \in \Sigma_a$  and every action  $a' \neq a$  we must satisfy that  $\mathbb{E}[u_r(a, \theta) \mid \sigma] \geq \mathbb{E}[u_r(a', \theta) \mid \sigma]$ , encoded by the first constraint. The other two constraints encode the feasibility of the scheme. Subject to these constraints, the objective is to maximize the expected utility of the sender. An optimal LP-solution  $x^*$  directly implies an optimal signaling scheme  $\varphi_P^*(\theta, \sigma) = x_{\theta, \sigma}^*$ . ◀

### 5.2 A $2n$ -Approximation Algorithm and Hardness

Going back to constrained persuasion with binary actions, we start by giving a simple  $2n$ -approximation algorithm. First, we give a useful benchmark for the optimal scheme.

► **Lemma 13.** *An optimal signaling scheme  $\varphi^*$  yields a sender utility of at most  $\min\{1, 2q_A\}$ .*

**Proof.** The upper bound of 1 is trivial.  $\varphi^*$  partitions the signal space into  $(\Sigma_A, \Sigma_R)$ , the accept and reject signals, respectively. The expected utility of the sender is

$$\sum_{\sigma \in \Sigma_A} \sum_{\theta \in N(\sigma)} \varphi^*(\theta, \sigma) \leq \sum_{\sigma \in \Sigma_A} \sum_{\theta \in N(\sigma) \cap \Theta_A} 2 \cdot \varphi^*(\theta, \sigma) \leq 2 \sum_{\theta \in \Theta_A} q_\theta = 2 \cdot q_A. \quad \blacktriangleleft$$

Our simple algorithm considers the  $m$  partitions with a single accept signal  $\Sigma_A = \{\sigma\}$ , for every  $\sigma \in \Sigma$ . For each such partition, the algorithm determines an optimal scheme and then picks the best one, among all  $m$  partitions. Instead of solving the LP of Proposition 12, given a proposed partition we proceed as follows. Assign as much probability mass from  $\Theta_A \cap N(\sigma)$  to  $\sigma$  and at most the same amount from  $\Theta_R \cap N(\sigma)$  – this ensures that  $\sigma$  is an accept signal. The remaining probability mass is assigned arbitrarily to other signals. Note that if this is impossible, there is no scheme that makes  $\sigma$  an accept signal.

► **Proposition 14.** *For constrained persuasion there is a  $2n$ -approximation algorithm that runs in polynomial time.*

**Proof.** Suppose  $\theta' \in \Theta_A$  is an acceptable state from which  $\varphi^*$  assigns the largest amount to accept signals, i.e.,  $\theta' = \arg \max_{\theta \in \Theta_A} \sum_{\sigma \in \Sigma_A \cap N(\theta)} \varphi^*(\theta, \sigma)$ . Clearly, the optimum accumulates on the accept signals at most  $n$  times this probability mass from the set of acceptable states, and at most the same from rejectable states. Hence,  $\sum_{\sigma \in \Sigma_A \cap N(\theta')} \varphi^*(\theta', \sigma) < q_{\theta'}$  is at least a  $1/(2n)$ -fraction of the optimal sender utility.

Consider the accept signals  $\Sigma_A$  in  $\varphi^*$  and any such signal  $\sigma' \in N(\theta') \cap \Sigma_A$ . When our algorithm checks the partition with  $\sigma'$  as the unique accept signal, it finds a feasible scheme, since the optimum scheme makes  $\sigma'$  an accept signal and the algorithm only assigns more probability from  $\Theta_A$  to  $\sigma'$ . The value of this solution is at least  $q_{\theta'}$ . ◀

In addition to this simple algorithm, we show a number of strong hardness results for constrained persuasion. The proofs of the following two theorems can be found in the full version.

► **Theorem 15.** *For any constant  $\varepsilon > 0$ , constrained persuasion is NP-hard to approximate within a factor of  $n^{1-\varepsilon}$ , even for instances with degree-2 states and degree-1 accepts.*

For instances with degree-1 rejects a similar result follows with a slightly different reduction.

► **Theorem 16.** *For any constant  $\varepsilon > 0$ , constrained persuasion is NP-hard to approximate within a factor of  $n^{1-\varepsilon}$ , even for instances with degree-1 rejects.*

### 5.3 Unique Accepts

In this section, we examine instances in which there is only a single acceptable state, for which we prove NP-hardness and give a PTAS. It will be convenient to state a lemma which allows us to get a better handle on the sender utility in an optimal signaling scheme for a given signal partition. This lemma will be helpful in both our hardness and algorithm analyses.

To state this lemma, we need some additional notation: for every subset  $\tilde{\Sigma} \subseteq \Sigma$ , we use  $\Theta_R(\tilde{\Sigma})$  to denote  $\{\theta \in \Theta_R \mid N(\theta) \subseteq \tilde{\Sigma}\}$ ; when  $\tilde{\Sigma} = \{\sigma\}$  is a singleton, we write  $\Theta_R(\sigma)$  in place of  $\Theta_R(\{\sigma\})$  for brevity. Moreover, let  $N(\tilde{\Sigma})$  denote  $\bigcup_{\sigma \in \tilde{\Sigma}} N(\sigma)$ . The lemma can now be stated as follows.

► **Lemma 17.** *Suppose that there exists a unique accept state  $\theta_a$ . For any partition  $P = (\Sigma_A, \Sigma_R)$  of the signal space such that  $\Sigma_A \neq \emptyset$ , we have*

1. *There exists a signaling scheme  $\varphi$  such that every signal in  $\Sigma_A$  is accepted and every signal in  $\Sigma_R$  is rejected by the receiver if and only if  $\Sigma_A \subseteq N(\theta_a)$  and  $\sum_{\theta \in \Theta_R(\Sigma_A)} q_\theta \leq q_{\theta_a}$ .*
2. *When the above condition holds, any optimal signaling scheme  $\varphi^*$  for the sender has utility equal to  $\min\{2q_{\theta_a}, \sum_{\theta \in N(\Sigma_A)} q_\theta\}$ , and, such a signaling scheme can be computed in polynomial time.*

We remark that the algorithm for finding  $\varphi^*$  in the above lemma is a simple greedy algorithm that tries to “put as much probability mass from rejectable states as possible” in  $\Sigma_A$  and then use the probability mass of the acceptable state  $\theta_a$  to “balance out” the mass from the rejectable states, so that eventually the signals in  $\Sigma_A$  are accepted. This is in contrast to the generic linear program-based algorithm in Proposition 12. The simpler greedy algorithm allows us to consider more concrete conditions and exactly compute the utility as stated in Lemma 17.

**Proof of Lemma 17.**

1. ( $\Rightarrow$ ) First, assume that there is such a signaling scheme  $\varphi$ . Clearly, every signal not in  $N(\theta_a)$  must be rejected, which implies that  $\Sigma_A \subseteq N(\theta_a)$ . Furthermore, for all  $\sigma \in \Sigma_A$ , we must have  $\varphi(\theta_a, \sigma) \geq \sum_{\theta \in N(\sigma) \cap \Theta_R} \varphi(\theta, \sigma)$ . Summing up over all  $\sigma \in \Sigma_A$  gives

$$\begin{aligned} q_{\theta_a} &\geq \sum_{\sigma \in \Sigma_A} \sum_{\theta \in N(\sigma) \cap \Theta_R} \varphi(\theta, \sigma) \\ &\geq \sum_{\sigma \in \Sigma_A} \sum_{\theta \in \Theta_R(\Sigma_A)} \varphi(\theta, \sigma) = \sum_{\theta \in \Theta_R(\Sigma_A)} \sum_{\sigma \in \Sigma_A} \varphi(\theta, \sigma) = \sum_{\theta \in \Theta_R(\Sigma_A)} q_{\theta}. \end{aligned}$$

- ( $\Leftarrow$ ) Assume that  $\emptyset \neq \Sigma_A \subseteq N(\theta_a)$  and  $\sum_{\theta \in \Theta_R(\Sigma_A)} q_{\theta} \leq q_{\theta_a}$ . We may construct a desired signaling scheme  $\varphi$  as follows. First, we assign  $\varphi(\theta, \sigma)$  arbitrarily for all  $\theta \in \Theta_R(\Sigma_A)$ . Then, we assign  $\varphi(\theta_a, \sigma)$  such that  $\varphi(\theta_a, \sigma) = 0$  for all  $\sigma \notin \Sigma_A$  and that  $\varphi(\theta_a, \sigma) \geq \sum_{\theta \in \Theta_R(\Sigma_A)} \varphi(\theta, \sigma)$  for all  $\sigma \in \Sigma_A$ . The former is possible because  $\Sigma_A \neq \emptyset$  and the latter possible because  $\sum_{\theta \in \Theta_R(\Sigma_A)} q_{\theta} \leq q_{\theta_a}$ . Finally, for each  $\theta \in \Theta_R \setminus \Theta_R(\Sigma_A)$ , assign  $\varphi(\theta, \sigma) = 0$  for all  $\sigma \in \Sigma_A$ . It is straightforward from the construction that this  $\varphi$  is a desired signaling scheme.
2. First, we will show that any signaling scheme  $\varphi$  has utility at most  $\min\{2q_{\theta_a}, \sum_{\theta \in N(\Sigma_A)} q_{\theta}\}$  for the sender. Observe that the upper bound  $2q_{\theta_a}$  follows trivially from Lemma 13. Thus, it suffices for us to prove that the utility is at most  $\sum_{\theta \in N(\Sigma_A)} q_{\theta}$ . To do so, let us rearrange the utility as follows:

$$\sum_{\sigma \in \Sigma_A} \sum_{\theta \in N(\sigma)} \varphi(\theta, \sigma) \leq \sum_{\theta \in N(\Sigma_A)} \sum_{\sigma \in N(\theta)} \varphi(\theta, \sigma) = \sum_{\theta \in N(\Sigma_A)} q_{\theta}.$$

Finally, we construct a signaling scheme  $\varphi^*$  with utility equal to  $\min\{2q_{\theta_a}, \sum_{\theta \in N(\Sigma_A)} q_{\theta}\}$ . The algorithm is a modification of the algorithm from the first part, and it works in four steps:

- For every  $\theta \in \Theta_R(\Sigma_A)$ , assign  $\varphi(\theta, \sigma)$  arbitrarily.
- For every  $\theta \in (N(\Sigma_A) \cap \Theta_R) \setminus \Theta_R(\Sigma_A)$ , assign  $\varphi(\theta, \sigma)$  so that

$$\sum_{\sigma \in \Sigma_A} \sum_{\theta \in N(\sigma) \cap \Theta_R} \varphi(\theta, \sigma) = \min\{q_{\theta_a}, \sum_{\theta \in N(\Sigma_A) \cap \Theta_R} q_{\theta}\}.$$

Note that this step is possible because  $\sum_{\theta \in \Theta_R(\Sigma_A)} q_{\theta} \leq q_{\theta_a}$ .

- Assign  $\varphi(\theta_a, \sigma)$  so that  $\varphi(\theta_a, \sigma) = 0$  for all  $\sigma \notin \Sigma_A$ , and that

$$\varphi(\theta_a, \sigma) \geq \sum_{\theta \in N(\sigma) \cap \Theta_R} \varphi(\theta, \sigma)$$

for all  $\sigma \in \Sigma_A$ . Note that this is possible because, from the previous step, we must have  $\sum_{\sigma \in \Sigma_A} \sum_{\theta \in N(\sigma) \cap \Theta_R} \varphi(\theta, \sigma) \leq q_{\theta_a}$ .

- All other remaining assignments are made arbitrarily in order to turn  $\varphi$  into a feasible signaling scheme.

It is straightforward to check that  $\varphi^*$  is the desired signaling scheme with utility equal to  $q_{\theta_a} + \min\{q_{\theta_a}, \sum_{\theta \in N(\Sigma_A) \cap \Theta_R} q_{\theta}\} = \min\{2q_{\theta_a}, \sum_{\theta \in N(\Sigma_A)} q_{\theta}\}$ .  $\blacktriangleleft$

With Lemma 17 ready, we now prove NP-hardness of the problem.

► **Theorem 18.** *Constrained persuasion with unique accepts is NP-hard.*



**Proof.** We reduce from the MAX-K-VERTEX-COVER problem, where we have a graph  $G = (V, E)$ . The goal is to choose a set  $V'$  of  $k$  vertices in order to maximize the number of edges incident to at least one vertex in  $V'$ . For every vertex  $v \in V$ , let  $E(v)$  be the set of incident edges, then we try to pick a subset  $V'$  of  $k$  vertices to maximize  $|\bigcup_{v \in V'} E(v)|$ .

For each edge  $e \in E$ , we introduce a rejectable state  $\theta_e$  with  $q_{\theta_e} = \frac{1}{(|V|+k)(|E|+1)+2|E|}$ . For each vertex  $v$  we introduce a signal  $\sigma_v$ . The graph  $H$  between states and signals expresses the incident property of edges and vertices. In addition, for each signal  $\sigma$ , we introduce an auxiliary rejectable states that have  $\sigma$  as their unique signal. Each auxiliary state  $\theta$  has  $q_\theta = \frac{|E|+1}{(|V|+k)(|E|+1)+2|E|}$ . Finally, the unique acceptable state  $\theta_a$  is incident to all signals and has probability

$$q_{\theta_a} = \frac{k(|E|+1) + |E|}{(|V|+k)(|E|+1) + 2|E|}.$$

From Lemma 17, the optimal signaling scheme has sender utility equal to

$$\max_{\Sigma_A} \min \left\{ 2q_{\theta_a}, \sum_{\theta \in N(\Sigma_A)} q_\theta \right\},$$

where the maximum is over non-empty  $\Sigma_A \subseteq \Sigma$  such that  $\sum_{\theta \in \Theta_R(\Sigma_A)} q_\theta \leq q_{\theta_a}$ . Notice that, in our construction, this condition is satisfied iff  $|\Sigma_A| \leq k$ . This means that  $\Sigma_A = \{\sigma_v\}_{v \in V'}$  for some subset  $V'$  of size at most  $k$ . It is also not hard to see that

$$\min \left\{ 2q_{\theta_a}, \sum_{\theta \in N(\Sigma_A)} q_\theta \right\} = \sum_{\theta \in N(\Sigma_A)} q_\theta = \frac{(|V'|+k)(|E|+1) + |\bigcup_{v \in V'} E(v)|}{(|V|+k)(|E|+1) + 2|E|}.$$

In other words, the utility is maximized iff  $V'$  is an optimal solution to the instance of MAX-K-VERTEX-COVER. Since the latter is NP-hard, we can conclude that constrained persuasion with unique accepts is also NP-hard.  $\blacktriangleleft$

We next give a PTAS for the problem. Before we formalize our PTAS, let us give an informal intuition. Observe that the condition in Lemma 17 implies that  $q_{\theta_a} \geq \sum_{\sigma \in \Sigma_A} \left( \sum_{\theta \in \Theta_R(\sigma)} q_\theta \right)$ . This latter constraint is a *knapsack constraint*. One generic strategy to solve knapsack problems is to first brute-force enumerate all possibilities of selecting “heavy items”, which in our case are the signals with large  $\sum_{\theta \in \Theta_R(\sigma)} q_\theta$ . Then, use a simple greedy algorithm for the remaining “light items”. Our PTAS follows this blueprint. However, since neither our constraints nor our objective function are exactly the same as in knapsack problems, we cannot use results from there directly and have to carefully argue the approximation guarantee ourselves.

► **Theorem 19.** *For constrained persuasion with unique accepts, for every fixed  $\varepsilon \in (0, 1]$ , Algorithm 1 runs in time  $m^{O(1/\varepsilon)} n^{O(1)}$  and outputs a  $(1 + \varepsilon)$  approximate solution.*

**Proof.** It is clear that our algorithm runs in time  $m^{O(1/\varepsilon)} n^{O(1)}$ . Let  $\varphi^*$  be any optimal signaling scheme, with utility OPT for the sender. We prove that the utility of  $\varphi_{\text{ALG}}$  is at least  $(1 - 0.5\varepsilon)\text{OPT}$ .

Without loss of generality we assume that the utility of  $\varphi^*$  is non-zero. Now, let  $(\Sigma_A^*, \Sigma_R^*)$  denote the signal partition of  $\varphi^*$ ; since the utility of  $\varphi^*$  is non-zero, we must have  $\Sigma_A^* \neq \emptyset$ . Furthermore, the first item of Lemma 17 implies that  $\Sigma_A^* \cap \Sigma_{\geq \varepsilon}$  must be of size at most  $1/\varepsilon$ . As a result, our algorithm must consider  $S = (\Sigma_A^* \cap \Sigma_{\geq \varepsilon})$  in the for-loop (3). For this particular  $S$ , let  $T'$  denote the largest  $T$  for which Line (6) is executed. We next consider two cases, based on whether or not we have  $T' = S \cup (\Sigma_{< \varepsilon} \cap N(\theta_a))$ .

■ **Algorithm 1** A PTAS for constrained persuasion with unique accepts.

---

**Input:** Graphs  $H$  with a single acceptable state  $\theta_a$ , and  $\varepsilon > 0$ .

- 1 Let  $\Sigma_{\geq \varepsilon}$  be the set of all signals  $\sigma \in \Sigma$  such that  $\sum_{\theta \in \Theta_R(\sigma)} q_\theta \geq \varepsilon q_{\theta_a}$ . Then, let  $\Sigma_{< \varepsilon} = \Sigma \setminus \Sigma_{\geq \varepsilon}$ ;
- 2 Let  $\varphi_{\text{ALG}}$  be an arbitrary signaling scheme;
- 3 **for** every (possibly empty) subset  $S \subseteq \Sigma_{\geq \varepsilon}$  of size at most  $1/\varepsilon$  **do**
- 4     Let  $T = S$ ;
- 5     **while**  $\sum_{\theta \in \Theta_R(T)} q_\theta \leq q_{\theta_a}$  **do**
- 6         If the utility of  $\varphi_{\text{ALG}}$  is less than  $\min\{2q_{\theta_a}, \sum_{\theta \in N(T)} q_\theta\}$ , then let  $\varphi_{\text{ALG}}$  be the optimal signaling scheme consistent with signaling partition  $\Sigma_A = T$ , which can be computed in polynomial time due to Lemma 17 ;
- 7         If  $T = \Sigma_{< \varepsilon} \cap N(\theta_a)$ , break from the loop;
- 8         Otherwise, add an arbitrary signal from  $(\Sigma_{< \varepsilon} \cap N(\theta_a)) \setminus T$  to  $T$ ;
- 9     **end**
- 10 **end**

**Output:**  $\varphi_{\text{ALG}}$ .

---

- Case I:  $T' = S \cup (\Sigma_{< \varepsilon} \cap N(\theta_a))$ . Notice that  $T' \supseteq \Sigma_A^*$ . Lemma 17, implies that the utility of  $\varphi_{\text{ALG}}$  must be at least OPT.
- Case II:  $T' \neq S \cup (\Sigma_{< \varepsilon} \cap N(\theta_a))$ . This means that there exists a signal  $\sigma^* \in (\Sigma_{< \varepsilon} \cap N(\theta_a))$  whose addition to  $T'$  breaks the condition of the while-loop (5), i.e.,  $q_{\theta_a} < \sum_{\theta \in \Theta_R(T' \cup \{\sigma^*\})} q_\theta$ . The right hand side of this inequality is equal to

$$\begin{aligned}
 \sum_{\substack{\theta \in \Theta_R \\ N(\theta) \subseteq (T' \cup \{\sigma^*\})}} q_\theta &\leq \sum_{\substack{\theta \in \Theta_R \\ N(\theta) \cap T' \neq \emptyset}} q_\theta + \sum_{\substack{\theta \in \Theta_R \\ N(\theta) = \{\sigma^*\}}} q_\theta \\
 &= \sum_{\theta \in N(T') \cap \Theta_R} q_\theta + \sum_{\theta \in \Theta_R(\sigma^*)} q_\theta \\
 &< \sum_{\theta \in N(T') \cap \Theta_R} q_\theta + \varepsilon q_{\theta_a} ,
 \end{aligned}$$

where the last inequality since  $\sigma$  belongs to  $\Sigma_{< \varepsilon}$ . Combining the two inequalities we have

$$\sum_{\theta \in N(T') \cap \Theta_R} q_\theta > (1 - \varepsilon) q_{\theta_a} . \tag{9}$$

On the other hand, from Lemma 17, when we execute line Line (6) for  $T = T'$ , it must result in a signaling scheme of utility

$$\min \left\{ 2q_{\theta_a}, \sum_{\theta \in N(T')} q_\theta \right\} = \min \left\{ 2q_{\theta_a}, q_{\theta_a} + \sum_{\theta \in N(T') \cap \Theta_R} q_\theta \right\} \stackrel{(9)}{>} (2 - \varepsilon) q_{\theta_a} ,$$

which is at least  $(1 - 0.5\varepsilon)\text{OPT}$  due to Lemma 13.

Hence, we can conclude that our algorithm always outputs a signaling scheme with sender utility at least  $(1 - 0.5\varepsilon)\text{OPT}$ . In other words, its approximation ratio is at most  $\frac{1}{1 - 0.5\varepsilon} \leq 1 + \varepsilon$ . ◀

---

References

---

- 1 Farid Alizadeh. Interior point methods in semidefinite programming with applications to combinatorial optimization. *SIAM J. Opt.*, 5(1):13–51, 1995.
- 2 Per Austrin. Balanced MAX 2-SAT might not be the hardest. In *Proc. 39th Symp. Theor. Comput. (STOC)*, pages 189–197, 2007.
- 3 Per Austrin. Towards sharp inapproximability for any 2-CSP. *SIAM J. Comput.*, 39(6):2430–2463, 2010.
- 4 Jesse Bull and Joel Watson. Hard evidence and mechanism design. *Games Econom. Behav.*, 58:75–93, 2007.
- 5 Gabriel Carroll and Georgy Egorov. Strategic communication with minimal verification. *Econometrica*, 87(6):1867–1892, 2019.
- 6 Shaddin Dughmi. On the hardness of designing public signals. *Games Econom. Behav.*, 118:609–625, 2019.
- 7 Shaddin Dughmi, David Kempe, and Ruixin Qiang. Persuasion with limited communication. In *Proc. 17th Conf. Economics and Computation (EC)*, pages 663–680, 2016.
- 8 Shaddin Dughmi, Rad Niazadeh, Alexandros Psomas, and Matthew Weinberg. Persuasion and incentives through the lens of duality. In *Proc. 15th Int. Conf. Web & Internet Econom. (WINE)*, pages 142–155, 2019.
- 9 Shaddin Dughmi and Haifeng Xu. Algorithmic Bayesian persuasion. In *Proc. 48th Symp. Theor. Comput. (STOC)*, pages 412–425, 2016.
- 10 Shaddin Dughmi and Haifeng Xu. Algorithmic persuasion with no externalities. In *Proc. 18th Conf. Economics and Computation (EC)*, pages 351–368, 2017.
- 11 Yuval Emek, Michal Feldman, Iftah Gamzu, Renato PaesLeme, and Moshe Tennenholtz. Signaling schemes for revenue maximization. *ACM Trans. Econom. Comput.*, 2(2):1–19, 2014.
- 12 Uriel Feige and Michel Goemans. Approximating the value of two power proof systems, with applications to MAX 2-SAT and MAX DI-CUT. In *Proc. 3rd Israel Symp. Theor. Comput. Syst.*, pages 182–189, 1995.
- 13 Jacob Glazer and Ariel Rubinstein. On optimal rules of persuasion. *Econometrica*, 72(6):1715–1736, 2004.
- 14 Jacob Glazer and Ariel Rubinstein. A study in the pragmatics of persuasion: a game theoretical approach. *Theor. Econom.*, 1(4):395–410, 2006.
- 15 Michel Goemans and David Williamson. .879-approximation algorithms for MAX CUT and MAX 2SAT. In *Proc. 26th Symp. Theor. Comput. (STOC)*, pages 422–431, 1994.
- 16 Ronen Gradwohl, Niklas Hahn, Martin Hoefer, and Rann Smorodinsky. Algorithms for persuasion with limited communication, 2020. CoRR abs/2007.12489. [arXiv:2007.12489](https://arxiv.org/abs/2007.12489).
- 17 Niklas Hahn, Martin Hoefer, and Rann Smorodinsky. Prophet inequalities for bayesian persuasion. In *Proc. 29th Int. Joint Conf. Artificial Intelligence (IJCAI)*, pages 175–181, 2020.
- 18 Niklas Hahn, Martin Hoefer, and Rann Smorodinsky. The secretary recommendation problem. In *Proc. 21st Conf. Economics and Computation (EC)*, page 189, 2020.
- 19 Emir Kamenica and Matthew Gentzkow. Bayesian persuasion. *Amer. Econom. Rev.*, 101(6):2590–2615, 2011.
- 20 Subhash Khot and Rishi Saket. Hardness of bipartite expansion. In *Proc. 24th European Symp. Algorithms (ESA)*, pages 55:1–55:17, 2016.
- 21 Michael Lewin, Dror Livnat, and Uri Zwick. Improved rounding techniques for the MAX 2-SAT and MAX DI-CUT problems. In *Int. Conf. Integer Prog. Combinat. Opt. (IPCO)*, pages 67–82, 2002.
- 22 Prasad Raghavendra. Optimal algorithms and inapproximability results for every CSP? In *Proc. 40th Symp. Theor. Comput. (STOC)*, pages 245–254, 2008.
- 23 Prasad Raghavendra and David Steurer. How to round any CSP. In *Proc. 50th Symp. Foundations of Computer Science (FOCS)*, pages 586–594, 2009.
- 24 Itai Sher. Credibility and determinism in a game of persuasion. *Games Econom. Behav.*, 71(2):409–419, 2011.

### 3:20    **Algorithmic Persuasion with Evidence**

- 25    Itai Sher. Persuasion and dynamic communication. *Theor. Econom.*, 9(1):99–136, 2014.
- 26    Henrik Sjögren. *Rigorous analysis of approximation algorithms for MAX 2-CSP*. Skolan för datavetenskap och kommunikation, Kungliga Tekniska högskolan, 2009.
- 27    Joel Sobel. Giving and receiving advice. *Adv. Econom. Econometrics*, 1:305–341, 2013.

# The Complexity of Finding Fair Independent Sets in Cycles

Ishay Haviv

School of Computer Science, The Academic College of Tel Aviv-Yaffo, Tel Aviv, Israel

---

## Abstract

Let  $G$  be a cycle graph and let  $V_1, \dots, V_m$  be a partition of its vertex set into  $m$  sets. An independent set  $S$  of  $G$  is said to *fairly represent* the partition if  $|S \cap V_i| \geq \frac{1}{2} \cdot |V_i| - 1$  for all  $i \in [m]$ . It is known that for every cycle and every partition of its vertex set, there exists an independent set that fairly represents the partition (Aharoni et al., A Journey through Discrete Math., 2017). We prove that the problem of finding such an independent set is PPA-complete. As an application, we show that the problem of finding a monochromatic edge in a Schrijver graph, given a succinct representation of a coloring that uses fewer colors than its chromatic number, is PPA-complete as well. The work is motivated by the computational aspects of the “cycle plus triangles” problem and of its extensions.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Problems, reductions and completeness; Mathematics of computing  $\rightarrow$  Graph theory

**Keywords and phrases** Fair independent sets in cycles, the complexity class {PPA}, Schrijver graphs

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.4

**Related Version** A full version of the paper is available at <http://arxiv.org/abs/2011.01770>.

**Funding** Research supported in part by the Israel Science Foundation (grant No. 1218/20).

**Acknowledgements** We are grateful to Aris Filos-Ratsikas and Alexander Golovnev for helpful discussions.

## 1 Introduction

In 1986, Du, Hsu, and Hwang [19] conjectured that if a graph on  $3n$  vertices is the disjoint union of a Hamilton cycle of length  $3n$  and  $n$  pairwise vertex-disjoint triangles then its independence number is  $n$ . The conjecture has become known as the “cycle plus triangles” problem and has been strengthened by Erdős [20], who conjectured that such a graph is 3-colorable. Fleischner and Stiebitz [26] confirmed these conjectures in a strong form and proved, using an algebraic approach of Alon and Tarsi [6], that such a graph is in fact 3-choosable. Their proof can also be viewed as an application of Alon’s Combinatorial Nullstellensatz technique [4]. Slightly later, an alternative elementary proof of the 3-coloring result was given by Sachs [39]. However, none of these proofs supplies an efficient algorithm that given a graph on  $3n$  vertices whose set of edges is the disjoint union of a Hamilton cycle and  $n$  pairwise vertex-disjoint triangles finds a 3-coloring of the graph or an independent set of size  $n$ . Questions on the computational aspects of the problem were posed in several works over the years (see, e.g., [27, 5, 9, 1]).

A natural extension of the problem of Du et al. [19] is the following. Let  $G$  be a cycle and let  $V_1, \dots, V_m$  be a partition of its vertex set into  $m$  sets. We are interested in an independent set of  $G$  that (almost) *fairly represents* the given partition, that is, an independent set  $S$  of  $G$  satisfying  $|S \cap V_i| \geq \frac{1}{2} \cdot |V_i| - 1$  for all  $i \in [m] = \{1, \dots, m\}$ . The existence of such an independent set was proved in a recent work of Aharoni, Alon, Berger, Chudnovsky, Kotlar, Loeb, and Ziv [1]. For the special case where all the sets  $V_i$  are of size 3, the proof technique of Aharoni et al. [1] allowed them to show that there are *two* disjoint independent



© Ishay Haviv;

licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 4; pp. 4:1–4:14



Leibniz International Proceedings in Informatics

LIPIcs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

sets that fairly represent the partition, providing a new proof of a stronger form of the original conjecture of Du et al. [19]. The results of [1] were extended in a work of Alishahi and Meunier [3] who proved the following.

► **Theorem 1** ([3]). *Let  $G$  be a cycle on  $n$  vertices and let  $V_1, \dots, V_m$  be a partition of its vertex set into  $m$  sets. Suppose that  $n$  and  $m$  have the same parity. Then, there exist two disjoint independent sets  $S_1$  and  $S_2$  of  $G$  covering all vertices but one from each  $V_i$  such that for each  $j \in \{1, 2\}$ , it holds that  $|S_j \cap V_i| \geq \frac{1}{2} \cdot |V_i| - 1$  for all  $i \in [m]$ .*

As shown by Black et al. [10], analogues of Theorem 1 for paths and for partitions into sets of odd sizes can also be proved using the approach of Aharoni et al. [1].

It is interesting to mention that although the statements of Theorem 1 and of its aforementioned variants are purely combinatorial, all of their known proofs are based on tools from topology. The use of topological methods in combinatorics was initiated by Lovász [32] who applied the Borsuk-Ulam theorem [11] from algebraic topology to prove a conjecture of Kneser [31] on the chromatic number of Kneser graphs. For integers  $n \geq 2k$ , the *Kneser graph*  $K(n, k)$  is the graph whose vertices are all the  $k$ -subsets of  $[n]$  where two sets are adjacent if they are disjoint. It was proved in [32] that the chromatic number of  $K(n, k)$  is  $n - 2k + 2$ , a result that was strengthened and generalized by several researchers (see, e.g., [34, Chapter 3]). One such strengthening was obtained by Schrijver [40], who studied the subgraph of  $K(n, k)$  induced by the collection of all  $k$ -subsets of  $[n]$  with no two consecutive elements modulo  $n$ . This graph is denoted by  $S(n, k)$  and is commonly referred to as the *Schrijver graph*. It was proved in [40], again by a topological argument, that the chromatic number of  $S(n, k)$  is equal to that of  $K(n, k)$ . As for Theorem 1, the proof of Alishahi and Meunier [3] employs the Octahedral Tucker lemma that was applied by Matoušek [33] in an alternative proof of Kneser's conjecture and can be viewed as a combinatorial formulation of the Borsuk-Ulam theorem (see also [42]). The approach of Aharoni et al. [1] and of Black et al. [10], however, is based on a direct application of the chromatic number of the Schrijver graph. As before, these proofs are not constructive, in the sense that they do not suggest efficient algorithms for the corresponding search problems. Understanding the computational complexity of these problems is the main motivation for the current work.

In 1994, Papadimitriou [38] has initiated the study of the complexity of total search problems in view of the mathematical argument that lies at the existence proof of their solutions. Let TFNP be the complexity class, defined in [35], of total search problems in NP, that is, the class of search problems in which a solution is guaranteed to exist and can be verified in polynomial running-time. Papadimitriou has introduced in [38] several subclasses of TFNP, each of which consists of the total search problems that can be reduced to a problem that represents some mathematical argument. For example, the class PPA (Polynomial Parity Argument) corresponds to the simple fact that every graph with maximum degree 2 that has a vertex of degree 1 must have another degree 1 vertex. Hence, PPA is defined as the class of all problems in TFNP that can be efficiently reduced to the LEAF problem, in which given a succinct representation of a graph with maximum degree 2 and given a vertex of degree 1 in the graph, the goal is to find another such vertex. The class PPAD (Polynomial Parity Argument in Directed graphs) is defined similarly with respect to directed graphs. Another complexity class defined in [38] is PPP (Polynomial Pigeonhole Principle) whose underlying mathematical argument is the pigeonhole principle. Additional examples of complexity classes defined in this way are PLS (Polynomial Local Search) [30], CLS (Continuous Local Search) [15], and EOPL (End of Potential Line) [21].

The complexity class PPAD is known to perfectly capture the complexity of many important search problems. Notable examples of PPAD-complete problems are those associated with Sperner's lemma [38, 12], the Nash Equilibrium theorem [13, 14], the Envy-Free Cake

Cutting theorem [18], and the Hairy Ball theorem [29]. For PPA, the undirected analogue of PPAD, until recently there were not known complete problems that are “natural”, i.e., do not involve circuits and Turing machines in their definitions. In the last few years, the situation was changed following a breakthrough result of Filos-Ratsikas and Goldberg [23, 24], who proved that the Consensus Halving problem with inverse-polynomial precision parameter is PPA-complete (see also [25]) and used it to derive the PPA-completeness of the classical Splitting Necklace with two thieves and Discrete Sandwich problems. This was obtained building on the PPA-hardness, proved by Aisenberg, Bonet, and Buss [2], of the search problem associated with Tucker’s lemma. The variant of the problem that corresponds to the Octahedral Tucker lemma was suggested for study by Pálvölgyi [37] and proved to be PPA-complete by Deng, Feng, and Kulkarni [17]. Additional PPA-complete problems, related to the Combinatorial Nullstellensatz and the Chevalley-Waring theorem, were provided by Belovs et al. [8].

## 1.1 Our Contribution

The present work initiates the study of the complexity of finding independent sets that fairly represent a given partition of the vertex set of a cycle. It is motivated by the computational aspects of combinatorial existence statements, such as the “cycle plus triangles” conjecture of Du et al. [19] proved by Fleischner and Stiebitz [26] and its extensions by Aharoni et al. [1], Alishahi and Meunier [3], and Black et al. [10]. As mentioned before, the challenge of understanding the complexity of the corresponding search problems was explicitly raised by several authors, including Fleischner and Stiebitz [27], Alon [5], and Aharoni et al. [1]. In this work we demonstrate that this research avenue may illuminate interesting connections between this family of problems and the complexity class PPA.

We start by introducing the Fair Independent Set in Cycle Problem, which we denote by FAIR-IS-CYCLE and define as follows.

► **Definition 2** (Fair Independent Set in Cycle Problem). *In the FAIR-IS-CYCLE problem, the input consists of a cycle  $G$  and a partition  $V_1, \dots, V_m$  of its vertex set into  $m$  sets. The goal is to find an independent set  $S$  of  $G$  satisfying  $|S \cap V_i| \geq \frac{1}{2} \cdot |V_i| - 1$  for all  $i \in [m]$ .*

The existence of a solution to every input of FAIR-IS-CYCLE is guaranteed by a result of Aharoni et al. [1, Theorem 1.8]. Since such a solution can be verified in polynomial running-time, the total search problem FAIR-IS-CYCLE lies in the complexity class TFNP. We prove that the class PPA captures the complexity of the problem.

► **Theorem 3.** *The FAIR-IS-CYCLE problem is PPA-complete.*

In view of the “cycle plus triangles” problem of Du et al. [19], it would be interesting to understand the complexity of the FAIR-IS-CYCLE problem restricted to partitions into sets of size 3. While Theorem 3 immediately implies that this restricted problem lies in PPA, the question of determining its precise complexity remains open.

We proceed by considering the search problem associated with Theorem 1. In the Fair Splitting of Cycle Problem, denoted FAIR-SPLIT-CYCLE, we are given a cycle and a partition of its vertex set and the goal is to find *two* disjoint independent sets that fairly represent the partition and cover all vertices but one from every part of the partition. We define below an approximation version of this problem, in which the fairness requirement is replaced with the relaxed notion of  $\varepsilon$ -fairness, where the independent sets should include at least  $\frac{1}{2} - \varepsilon$  fraction of the vertices of every part.



► **Definition 4** (Approximate Fair Splitting of Cycle Problem). *In the  $\varepsilon$ -FAIR-SPLIT-CYCLE problem with parameter  $\varepsilon \geq 0$ , the input consists of a cycle  $G$  on  $n$  vertices and a partition  $V_1, \dots, V_m$  of its vertex set into  $m$  sets, such that  $n$  and  $m$  have the same parity. The goal is to find two disjoint independent sets  $S_1$  and  $S_2$  of  $G$  covering all vertices but one from each  $V_i$  such that for each  $j \in \{1, 2\}$ , it holds that  $|S_j \cap V_i| \geq (\frac{1}{2} - \varepsilon) \cdot |V_i| - 1$  for all  $i \in [m]$ . For  $\varepsilon = 0$ , the problem is denoted by FAIR-SPLIT-CYCLE.*

The existence of a solution to every input of  $\varepsilon$ -FAIR-SPLIT-CYCLE, already for  $\varepsilon = 0$ , is guaranteed by Theorem 1 proved in [3]. For  $\varepsilon = 0$ , it can be seen that FAIR-SPLIT-CYCLE is at least as hard as FAIR-IS-CYCLE. Yet, it turns out that FAIR-SPLIT-CYCLE lies in PPA and is thus also PPA-complete.

► **Theorem 5.** *The FAIR-SPLIT-CYCLE problem is PPA-complete.*

For the approximation version of the problem, we provide the following hardness result.

► **Theorem 6.** *There exists a constant  $\varepsilon > 0$  for which the  $\varepsilon$ -FAIR-SPLIT-CYCLE problem is PPAD-hard.*

We finally consider the complexity of the SCHRIJVER problem. In this problem we are given a succinct representation of a coloring of the Schrijver graph  $S(n, k)$  with  $n - 2k + 1$  colors, which is one less than its chromatic number [40], and the goal is to find a monochromatic edge (see Definition 16). The study of the SCHRIJVER problem is motivated by a question raised by Deng et al. [17] regarding the complexity of the analogue problem for Kneser graphs. Note that the latter is not harder than the SCHRIJVER problem, because  $S(n, k)$  is a subgraph of  $K(n, k)$  with the same chromatic number. As an application of our Theorem 3, we prove the following.

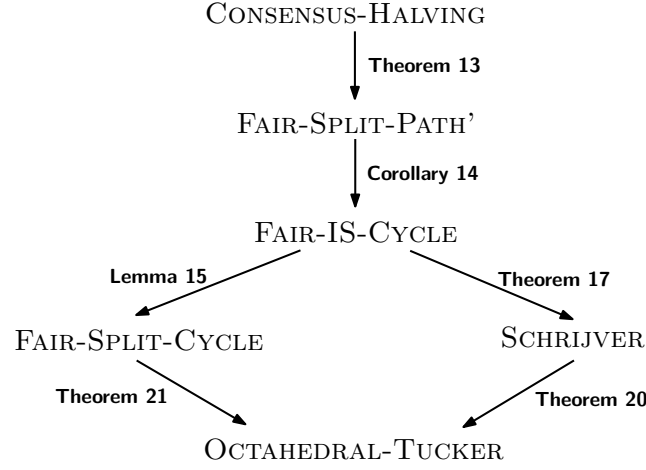
► **Theorem 7.** *The SCHRIJVER problem is PPA-complete.*

## 1.2 Overview of Proofs

To obtain our results we present a chain of reductions, as described in Figure 1. Our starting point is the Consensus Halving problem with precision parameter  $\varepsilon$ , in which given a collection of  $m$  probability measures on the interval  $[0, 1]$  the goal is to partition the interval into two pieces using relatively few cuts, so that each of the measures has the same mass on the two pieces up to an error of  $\varepsilon$  (see Definition 8). It is known that every input to this problem has a solution with at most  $m$  cuts even for  $\varepsilon = 0$  [41] (see also [28, 7]). The problem of finding a solution is PPA-hard when  $\varepsilon$  is inverse-polynomial in  $m$  [23, 24, 25] and PPAD-hard when  $\varepsilon$  is some positive constant [22].

In Section 2, we reduce the Consensus Halving problem to an intermediate variant of the  $\varepsilon$ -FAIR-SPLIT-CYCLE problem, which we call  $\varepsilon$ -FAIR-SPLIT-PATH' (see Definition 12). Then, we use this reduction to obtain our hardness results for the FAIR-IS-CYCLE and FAIR-SPLIT-CYCLE problems. The reduction borrows a discretization argument that was used in [23] to reduce the Consensus Halving problem to the Splitting Necklace problem with two thieves. This argument enables us to transform a Consensus Halving instance into a path and a partition of its vertex set, for which the goal is to partition the path using relatively few cuts into two parts, each of which contains roughly half of the vertices of every set in the partition. In order to relate this problem to independent sets that fairly represent the partition, we need an additional simple trick. Between every two consecutive vertices of the path we add a new vertex and put all the new vertices in a new set added to the partition





■ **Figure 1** Chain of reductions.

of the vertex set. We then argue, roughly speaking, that two disjoint independent sets in the obtained path, which fairly represent the partition and cover almost all of the vertices, can be used to obtain a solution to the original instance. The high-level idea is that those few vertices that are uncovered by the two independent sets can be viewed as cuts, and every path between two such vertices alternates between the two given independent sets. By construction, it means that only one of the two independent sets contains in such a path original vertices (that is, vertices that were not added in the last phase of the reduction), hence every such path can be naturally assigned to one of the two pieces required by the Consensus Halving problem. Combining our reduction with the known hardness results of Consensus Halving, we derive the PPA-hardness of FAIR-IS-CYCLE and FAIR-SPLIT-CYCLE and the PPAD-hardness of  $\varepsilon$ -FAIR-SPLIT-CYCLE for a constant  $\varepsilon > 0$ , as needed for Theorems 3, 5, and 6.

In Section 3, we introduce and study the SCHRIJVER problem. We reduce the FAIR-IS-CYCLE problem to the SCHRIJVER problem, implying that the latter is PPA-hard. The reduction follows an argument of Aharoni et al. [1] who used the chromatic number of the Schrijver graph [40] to prove the existence of the independent set required in FAIR-IS-CYCLE. Finally, employing arguments of Meunier [36] and Alishahi and Meunier [3], we reduce the SCHRIJVER and FAIR-SPLIT-CYCLE problems to the search problem associated with the Octahedral Tucker lemma (see Definition 18). Since it is known, already from [38], that this problem lies in PPA, we get that FAIR-IS-CYCLE, FAIR-SPLIT-CYCLE, and SCHRIJVER are all members of PPA, completing the proofs of Theorems 3, 5, and 7.

We remark that one could consider analogues of the FAIR-IS-CYCLE and FAIR-SPLIT-CYCLE problems for paths rather than for cycles and obtain similar results. We have chosen to focus here on the cycle setting, motivated by the computational aspects of the “cycle plus triangles” problem [19, 20, 26].

## 2 Fair Independent Sets in Cycles

In this section we prove our hardness results for the FAIR-IS-CYCLE and FAIR-SPLIT-CYCLE problems. We first recall the definition of the Consensus Halving problem and gather some of the hardness results known for it. Then, we present an efficient reduction from this problem to an intermediate problem, which is used to obtain the hardness results of Theorems 3, 5, and 6.

## 2.1 Consensus Halving

Consider the following variant of the Consensus Halving problem, denoted CON-HALVING.

► **Definition 8** (Consensus Halving Problem). *In the  $\varepsilon$ -CON-HALVING( $m, \ell$ ) problem with precision parameter  $\varepsilon = \varepsilon(m)$ , the input consists of  $m$  probability measures  $\mu_1, \dots, \mu_m$  on the interval  $I = [0, 1]$ , given by their density functions. The goal is to partition the interval  $I$  using at most  $\ell$  cuts into two (not necessarily connected) pieces  $I^+$  and  $I^-$ , so that for every  $i \in [m]$  it holds that  $|\mu_i(I^+) - \mu_i(I^-)| \leq \varepsilon$ .*

For  $\ell \geq m$ , every input of  $\varepsilon$ -CON-HALVING( $m, \ell$ ) has a solution even for  $\varepsilon = 0$  [41]. We state below two hardness results known for CON-HALVING. Here, a function on an interval is said to be *piecewise constant* if its domain can be partitioned into a finite set of intervals such that the function is constant on each of them. We refer to the intervals of the partition on which the function is nonzero as the *blocks* of the function.

► **Theorem 9** ([23, 24, 25]). *There exists an inverse-polynomial  $\varepsilon = \varepsilon(m)$  such that for every constant  $c \geq 0$ , the  $\varepsilon$ -CON-HALVING( $m, m + c$ ) problem, restricted to inputs with piecewise constant density functions with at most 2 blocks, is PPA-hard.*

► **Theorem 10** ([22]). *There exist an absolute constant  $\varepsilon > 0$  and a polynomial  $p$  such that for every constant  $c \geq 0$ , the  $\varepsilon$ -CON-HALVING( $m, m + c$ ) problem, restricted to inputs with piecewise constant density functions with at most  $p(m)$  blocks, is PPAD-hard.*

► **Remark 11.** We note that, as explained in [25], the constant  $c$  given in Theorems 9 and 10 can be replaced by  $m^{1-\alpha}$  for any constant  $\alpha > 0$ . This stronger hardness, however, is not required to obtain our results. We also note that our results do not rely on the fact that the hardness given in Theorem 9 holds for instances with density functions with at most 2 blocks, as proved in [25], rather than polynomially many blocks as was previously proved in [24].

## 2.2 The Main Reduction

To obtain our hardness results for the FAIR-IS-CYCLE and FAIR-SPLIT-CYCLE problems, we consider the following intermediate problem.

► **Definition 12.** *In the  $\varepsilon$ -FAIR-SPLIT-PATH' problem with parameter  $\varepsilon \geq 0$ , the input consists of a path  $G$  and a partition  $V_1, \dots, V_m$  of its vertex set into  $m$  sets such that  $|V_i|$  is odd for all  $i \in [m]$ . The goal is to find two disjoint independent sets  $S_1$  and  $S_2$  of  $G$  covering all but at most  $m$  of the vertices of  $G$  such that*

$$|S_1 \cap V_i| \in \left[ \left(\frac{1}{2} - \varepsilon\right) \cdot |V_i| - 1, \left(\frac{1}{2} + \varepsilon\right) \cdot |V_i| \right]$$

*for all  $i \in [m]$ . When  $\varepsilon = 0$ , the problem is denoted by FAIR-SPLIT-PATH'.*

Note that the  $\varepsilon$ -FAIR-SPLIT-PATH' problem differs from the  $\varepsilon$ -FAIR-SPLIT-CYCLE problem (see Definition 4) in the following respects: (a) The input graph is a path rather than a cycle, (b) an  $\varepsilon$ -fairness property is required only for the independent set  $S_1$  rather than for both  $S_1$  and  $S_2$ , (c) there is no requirement regarding the sets  $V_i$  to which the vertices that are uncovered by  $S_1$  and  $S_2$  belong, and (d) the sets  $V_i$  are required to be of odd sizes. Yet, it is easy to check that Theorem 1 implies that every instance of the  $\varepsilon$ -FAIR-SPLIT-PATH' problem has a solution already for  $\varepsilon = 0$ .

We turn to prove the following.

► **Theorem 13.** *Let  $p$  be a polynomial and suppose that  $\varepsilon = \varepsilon(m)$  is bounded from below by some inverse-polynomial in  $m$ . Then, the  $\varepsilon$ -CON-HALVING( $m, m+1$ ) problem, restricted to inputs with piecewise constant density functions with at most  $p(m)$  blocks, is polynomial-time reducible to the  $\frac{\varepsilon}{4}$ -FAIR-SPLIT-PATH' problem.*

**Proof.** Consider an instance of  $\varepsilon$ -CON-HALVING( $m, m+1$ ) consisting of  $m$  probability measures  $\mu_1, \dots, \mu_m$  on the interval  $I = [0, 1]$ , given by their piecewise constant density functions  $g_1, \dots, g_m$ , each of which has at most  $p(m)$  blocks. The reduction constructs an instance of  $\frac{\varepsilon}{4}$ -FAIR-SPLIT-PATH', namely, a path  $G$  and a partition  $V_1, \dots, V_{m+1}$  of its vertex set into  $m+1$  sets of odd sizes.

We start with a high-level description of the reduction. The reduction associates with every density function  $g_i$  a collection  $V_i$  of vertices located in the (at most  $p(m)$ ) intervals on which  $g_i$  is nonzero. To do so, we partition every block of  $g_i$  into sub-intervals such that the measure of  $\mu_i$  on each of them is  $\delta$ , where  $\delta > 0$  is some small parameter (assuming, for now, that the measure of  $\mu_i$  on every block is an integer multiple of  $\delta$ ). At the middle of every such sub-interval we locate a vertex and put it in  $V_i$ . Then, we construct a path  $G$  that alternates between the vertices of  $V_1 \cup \dots \cup V_m$  ordered according to their locations in  $I$  and additional vertices which we put in another set  $V_{m+1}$ . We also take care of the requirement that each  $|V_i|$  is odd.

The intuitive idea behind this reduction is the following. Suppose that we are given a solution to the constructed instance, i.e., two disjoint independent sets  $S_1$  and  $S_2$  of the path  $G$  covering all but  $m+1$  of the vertices such that  $S_1$  contains roughly half of the vertices of  $V_i$  for each  $i \in [m+1]$ . Observe that by removing from  $G$  the  $m+1$  vertices that *do not* belong to  $S_1 \cup S_2$ , we essentially get a partition of the vertices of  $S_1 \cup S_2$  into  $m+2$  paths. Since  $S_1$  and  $S_2$  are independent sets in  $G$ , it follows that each such path alternates between  $S_1$  and  $S_2$ . However, recalling that  $G$  alternates between  $V_1 \cup \dots \cup V_m$  and  $V_{m+1}$ , it follows that ignoring the vertices of  $V_{m+1}$ , each such path contains either only vertices of  $S_1$  or only vertices of  $S_2$ . Now, one can view the  $m+1$  locations of the vertices that do not belong to  $S_1 \cup S_2$  as cuts in the interval  $I$  which partition it into  $m+2$  sub-intervals, each of which includes vertices from either  $S_1$  or  $S_2$  (again, ignoring the vertices of  $V_{m+1}$ ). Let  $I^+$  and  $I^-$  be the pieces of  $I$  obtained from the sub-intervals that correspond to  $S_1$  and  $S_2$  respectively. Since the number of vertices from  $V_i$  in every path is approximately proportional to the measure of  $\mu_i$  in the corresponding sub-interval, it can be shown that the probability measure of  $\mu_i$  on  $I^+$  is approximately  $\frac{1}{2}$ . This yields that the probability measure  $\mu_i$  is approximately equal on the pieces  $I^+$  and  $I^-$ , as needed for the CON-HALVING( $m, m+1$ ) problem.

We turn to the formal description of the reduction. Define  $\delta = \frac{\varepsilon}{4 \cdot (2p(m) + m + 3)}$ . The reduction acts as follows.

1. For every  $i \in [m]$ , do the following:
  - We are given a partition of the interval  $I$  into intervals such that on at most  $p(m)$  of them the function  $g_i$  is equal to a nonzero value and is zero everywhere else. For every such interval, let  $\gamma$  denote the volume of  $g_i$  on it, and divide it into  $\lceil \gamma/\delta \rceil$  sub-intervals of volume  $\delta$  each, possibly besides the last one whose volume might be smaller. We refer to a sub-interval of volume smaller than  $\delta$  as an *imperfect* sub-interval. The number of imperfect sub-intervals associated with  $g_i$  is clearly at most  $p(m)$ . At the middle point of every sub-interval of  $g_i$ , locate a vertex and put it in the set  $V_i$ .
  - If the number of vertices in  $V_i$  is even, then add another vertex to  $V_i$  and locate it arbitrarily in  $I$ .
  - Note that, by  $\mu_i(I) = 1$ , we have
 
$$|V_i| \cdot \delta \in [1, 1 + (p(m) + 1) \cdot \delta]. \quad (1)$$

2. Consider the path on the vertices of  $V_1 \cup \dots \cup V_m$  ordered according to their locations in the interval  $I$ , breaking ties arbitrarily.
3. Add a new vertex before every vertex in this path, locate it at the middle of the sub-interval between its two adjacent vertices (where the first new vertex is located at 0), and put these new vertices in the set  $V_{m+1}$ . If the number of vertices in  $V_{m+1}$  is even then add one more vertex to the end of the path, locate it at 1, and put it in  $V_{m+1}$  as well. Denote by  $G$  the obtained path, and note that  $G$  alternates between  $V_1 \cup \dots \cup V_m$  and  $V_{m+1}$ .
4. The output of the reduction is the path  $G$  and the partition  $V_1, \dots, V_{m+1}$  of its vertex set  $V$  into  $m+1$  sets. By construction,  $|V_i|$  is odd for every  $i \in [m+1]$ .

It is easy to verify that the reduction can be implemented in polynomial running-time. Indeed, every density function  $g_i$  is piecewise constant with at most  $p(m)$  blocks, hence for every  $i \in [m]$  the number of vertices that the reduction defines for  $V_i$  is at most  $1/\delta + p(m) + 1$ , and the latter is polynomial in the input size because of the definition of  $\delta$  and the fact that  $\varepsilon$  is at least inverse-polynomial in  $m$ . The additional set  $V_{m+1}$  doubles the number of vertices, possibly with one extra vertex, preserving the construction polynomial in the input size.

We turn to prove the correctness of the reduction, that is, that a solution to the constructed instance of  $\frac{\varepsilon}{4}$ -FAIR-SPLIT-PATH' can be used to efficiently compute a solution to the original instance of  $\varepsilon$ -CON-HALVING( $m, m+1$ ). Suppose we are given a solution to  $\frac{\varepsilon}{4}$ -FAIR-SPLIT-PATH' for the path  $G$  and the partition  $V_1, \dots, V_{m+1}$  of its vertex set  $V$ . Such a solution consists of two disjoint independent sets  $S_1$  and  $S_2$  of  $G$  covering all but at most  $m+1$  of the vertices of  $G$  such that

$$|S_1 \cap V_i| \in \left[ \left( \frac{1}{2} - \frac{\varepsilon}{4} \right) \cdot |V_i| - 1, \left( \frac{1}{2} + \frac{\varepsilon}{4} \right) \cdot |V_i| \right] \quad (2)$$

for all  $i \in [m+1]$ . Put  $S_3 = V \setminus (S_1 \cup S_2)$ . It can be assumed that  $|S_3| = m+1$  (otherwise, remove some arbitrary vertices from  $S_2$ ). Denote the vertices of  $S_3$  by  $u_1, \dots, u_{m+1}$  ordered according to their order in  $G$ . Let  $P_1, \dots, P_{m+2}$  be the  $m+2$  paths obtained from  $G$  by removing the vertices of  $S_3$  (where some of the paths might be empty). Since  $S_1$  and  $S_2$  are independent sets, every path  $P_j$  alternates between  $S_1$  and  $S_2$ . By our construction, this implies that in every path  $P_j$  either the vertices of  $S_1$  are from  $V \setminus V_{m+1}$  and those of  $S_2$  are from  $V_{m+1}$ , or the vertices of  $S_2$  are from  $V \setminus V_{m+1}$  and those of  $S_1$  are from  $V_{m+1}$ . We define  $b_j = 1$  in the former case and  $b_j = 2$  in the latter. Thus, for every  $i \in [m]$ , the number of vertices of  $V_i$  that appear in the paths  $P_j$  with  $b_j = 1$  is precisely  $|S_1 \cap V_i|$ .

Now, let  $\beta_1, \dots, \beta_{m+1} \in I$  be the locations of the vertices  $u_1, \dots, u_{m+1}$  in the interval  $I$  as defined by the reduction. We interpret these locations as  $m+1$  cuts of the interval  $I$ . Set  $\beta_0 = 0$  and  $\beta_{m+2} = 1$ , and for every  $j \in [m+2]$ , let  $I_j$  denote the interval  $[\beta_{j-1}, \beta_j]$ . Consider the partition of  $I$  into two pieces  $I^+$  and  $I^-$ , where  $I^+$  includes all the parts  $I_j$  with  $b_j = 1$  and  $I^-$  includes all the parts  $I_j$  with  $b_j = 2$ . We claim that this partition, which is obtained using  $m+1$  cuts in  $I$ , forms a valid solution to the original instance of  $\varepsilon$ -CON-HALVING( $m, m+1$ ). To this end, we show that for every  $i \in [m]$  it holds that  $|\mu_i(I^+) - \frac{1}{2}| \leq \frac{\varepsilon}{2}$ , which is equivalent to  $|\mu_i(I^+) - \mu_i(I^-)| \leq \varepsilon$ .

Fix some  $i \in [m]$ . We turn to estimate the quantity  $\mu_i(I^+)$ , i.e., the total measure of  $\mu_i$  on the intervals  $I_j$  with  $b_j = 1$ . By our construction, every vertex of  $V_i$  corresponds to a sub-interval whose measure by  $\mu_i$  is  $\delta$  (except for at most  $p(m) + 1$  of them). Since the intervals of  $I^+$  correspond to the paths  $P_j$  whose vertices in  $V \setminus V_{m+1}$  are precisely the vertices of  $S_1 \setminus V_{m+1}$ , one would expect  $\mu_i(I^+)$  to measure the number of vertices in  $S_1 \cap V_i$ , with a contribution of  $\delta$  per every such vertex. This suggests an estimation of  $|S_1 \cap V_i| \cdot \delta$  for  $\mu_i(I^+)$ . However, several issues might prevent from this estimation to be accurate:

- The set  $V_i$  might include vertices that correspond to imperfect sub-intervals whose measure by  $\mu_i$  is smaller than  $\delta$ . Since there are at most  $p(m)$  such vertices in  $V_i$ , they can cause an error of at most  $p(m) \cdot \delta$  in the above estimation.
- To make sure that  $|V_i|$  is odd, the reduction might add one extra vertex to  $V_i$ . This might cause an error of at most  $\delta$  in the above estimation.
- The precise locations  $\beta_j$  of the cuts of  $I$  might fall inside sub-intervals that correspond to vertices of  $V_i$ . Since the sub-intervals that correspond to vertices of  $V_i$  are disjoint, every such cut can cause an error of at most  $\delta$  in the above estimation, and since there are  $m + 1$  cuts the error here is bounded by  $(m + 1) \cdot \delta$ .

We conclude that  $\mu_i(I^+)$  differs from the aforementioned estimation  $|S_1 \cap V_i| \cdot \delta$  by not more than  $(p(m) + m + 2) \cdot \delta$ . Combining (1) and (2), it can be verified that

$$\left| |S_1 \cap V_i| \cdot \delta - \frac{1}{2} \right| \leq \frac{\varepsilon}{4} + (p(m) + 1) \cdot \delta,$$

hence

$$\begin{aligned} \left| \mu_i(I^+) - \frac{1}{2} \right| &\leq \left| \mu_i(I^+) - |S_1 \cap V_i| \cdot \delta \right| + \left| |S_1 \cap V_i| \cdot \delta - \frac{1}{2} \right| \\ &\leq (p(m) + m + 2) \cdot \delta + \frac{\varepsilon}{4} + (p(m) + 1) \cdot \delta \\ &= \frac{\varepsilon}{4} + (2p(m) + m + 3) \cdot \delta = \frac{\varepsilon}{2}, \end{aligned}$$

where the last equality holds by the definition of  $\delta$ . This completes the proof.  $\blacktriangleleft$

### 2.3 Hardness of Fair-IS-Cycle and Fair-Split-Cycle

Equipped with Theorem 13, we are ready to derive the hardness of the FAIR-IS-CYCLE and FAIR-SPLIT-CYCLE problems (see Definitions 2 and 4).

► **Corollary 14.** *The FAIR-IS-CYCLE problem is PPA-hard.*

**Proof.** By Theorem 9, the  $\varepsilon$ -CON-HALVING( $m, m + 1$ ) problem is PPA-hard for input density functions that are piecewise constant with at most 2 blocks, where  $\varepsilon = \varepsilon(m)$  is inverse-polynomial. By Theorem 13, this problem is polynomial-time reducible to the  $\frac{\varepsilon}{4}$ -FAIR-SPLIT-PATH' problem, implying that FAIR-SPLIT-PATH', with  $\varepsilon = 0$ , is PPA-hard. Hence, to prove the corollary, it suffices to show that FAIR-SPLIT-PATH' is polynomial-time reducible to FAIR-IS-CYCLE.

Consider an instance of FAIR-SPLIT-PATH', that is, a path  $G$  on  $n$  vertices and a partition  $V_1, \dots, V_m$  of its vertex set into  $m$  sets such that  $|V_i|$  is odd for all  $i \in [m]$ . The reduction simply returns the cycle  $G'$ , obtained from the path  $G$  by connecting its endpoints by an edge, and the same partition  $V_1, \dots, V_m$  of its vertex set. For correctness, suppose that we are given a solution to this instance of FAIR-IS-CYCLE, i.e., an independent set  $S_1$  of  $G'$  satisfying  $|S_1 \cap V_i| \geq \frac{1}{2} \cdot |V_i| - 1$  for all  $i \in [m]$ . Since each  $|V_i|$  is odd, it can be assumed that  $|S_1 \cap V_i| = \frac{1}{2} \cdot (|V_i| - 1)$  for all  $i \in [m]$  (by removing some vertices from  $S_1$  if needed), implying that

$$|S_1| = \sum_{i=1}^m |S_1 \cap V_i| = \frac{1}{2} \cdot \sum_{i=1}^m (|V_i| - 1) = \frac{n - m}{2}.$$

For every vertex of  $S_1$  consider the vertex that follows it in the cycle  $G'$  (say, oriented clockwise), and let  $S_2$  be the set of vertices that follow those of  $S_1$ . Since  $S_1$  is an independent

## 4:10 The Complexity of Finding Fair Independent Sets in Cycles

set in  $G'$ , we get that  $S_2$  is another independent set in  $G'$  which is disjoint from  $S_1$  and has the same size. We obtain that

$$|S_1 \cup S_2| = |S_1| + |S_2| = 2 \cdot \frac{n-m}{2} = n-m,$$

hence  $S_1$  and  $S_2$  are two disjoint independent sets of  $G'$  covering all but  $m$  of its vertices. In particular,  $S_1$  and  $S_2$  are independent sets in the path  $G$ , and as such, they form a valid solution to the FAIR-SPLIT-PATH' instance. This solution can clearly be constructed in polynomial running-time given  $S_1$ , completing the proof. ◀

The following simple lemma allows us to derive the PPA-hardness of FAIR-SPLIT-CYCLE.

► **Lemma 15.** *The FAIR-IS-CYCLE problem is polynomial-time reducible to FAIR-SPLIT-CYCLE.*

**Proof.** Consider an instance of FAIR-IS-CYCLE, that is, a cycle  $G$  on  $n$  vertices and a partition  $V_1, \dots, V_m$  of its vertex set into  $m$  sets. If  $n$  and  $m$  have the same parity then the reduction returns the input as is. Otherwise, there exists some  $i \in [m]$  for which the size of  $V_i$  is even. In this case, the reduction adds to the cycle  $G$  a new vertex located between two arbitrary consecutive vertices and puts it in  $V_i$ . Now, the number of vertices and the number of sets in the partition have the same parity, so the reduction can output the obtained cycle and partition.

A solution to the constructed instance of FAIR-SPLIT-CYCLE involves two disjoint independent sets that fairly represent the partition. Clearly, at least one of the sets does not include the two neighbors of the vertex that was possibly added to  $G$ . Letting  $S$  denote the set of vertices of  $G$  in this set, we get that  $S$  is independent in  $G$ , and it is easy to check that  $|S \cap V_i| \geq \frac{1}{2} \cdot |V_i| - 1$  for all  $i \in [m]$ , so we are done. ◀

We end this section with a proof of Theorem 6.

**Proof of Theorem 6.** By Theorem 10, the  $\varepsilon$ -CON-HALVING( $m, m+1$ ) problem is PPAD-hard for input density functions that are piecewise constant with at most  $p(m)$  blocks, where  $p$  is a polynomial and  $\varepsilon$  is a positive constant. Applying Theorem 13, we get that the  $\frac{\varepsilon}{4}$ -FAIR-SPLIT-PATH' problem is PPAD-hard. To complete the proof, we show that for every  $\varepsilon \geq 0$  the  $\varepsilon$ -FAIR-SPLIT-PATH' problem is polynomial-time reducible to the  $\varepsilon$ -FAIR-SPLIT-CYCLE problem.

Consider again the reduction that given a path  $G$  and a partition  $V_1, \dots, V_m$  of its vertex set into sets of odd sizes returns the cycle  $G'$ , obtained from the path  $G$  by connecting its endpoints by an edge, and the same partition  $V_1, \dots, V_m$ . Since the sets of the partition have odd sizes, it follows that the number of vertices and the number of sets in the partition have the same parity, hence the reduction provides an appropriate instance of the  $\varepsilon$ -FAIR-SPLIT-CYCLE problem.

For correctness, consider a solution to the constructed instance, i.e., two disjoint independent sets  $S_1$  and  $S_2$  of  $G'$  covering all vertices but one from each part  $V_i$  such that for each  $j \in \{1, 2\}$ , it holds that  $|S_j \cap V_i| \geq (\frac{1}{2} - \varepsilon) \cdot |V_i| - 1$  for all  $i \in [m]$ . We claim that  $S_1$  and  $S_2$  form a valid solution to the original  $\varepsilon$ -FAIR-SPLIT-PATH' instance. Indeed, an independent set in  $G'$  is also an independent set in  $G$ . In addition, the set  $S_1$  satisfies  $|S_1 \cap V_i| \in [(\frac{1}{2} - \varepsilon) \cdot |V_i| - 1, (\frac{1}{2} + \varepsilon) \cdot |V_i|]$  for all  $i \in [m]$ , where the upper bound holds because

$$|S_1 \cap V_i| = |V_i| - |S_2 \cap V_i| - 1 \leq |V_i| - \left( (\frac{1}{2} - \varepsilon) \cdot |V_i| - 1 \right) - 1 = (\frac{1}{2} + \varepsilon) \cdot |V_i|.$$

This completes the proof. ◀

### 3 The Schrijver Problem

In this section we introduce and study the SCHRIJVER problem, a natural analogue of the Kneser problem defined by Deng et al. [17].

We start with some definitions. A set  $A \subseteq [n]$  is said to be *stable* if it does not contain two consecutive elements modulo  $n$  (that is, if  $i \in A$  then  $i + 1 \notin A$ , and if  $n \in A$  then  $1 \notin A$ ). In other words, a stable subset of  $[n]$  is an independent set in the cycle on  $n$  vertices with the numbering from 1 to  $n$  along the cycle. For integers  $n \geq 2k$ , let  $\binom{[n]}{k}_{\text{stab}}$  denote the collection of all stable  $k$ -subsets of  $[n]$ . Recall that the Schrijver graph  $S(n, k)$  is the graph on the vertex set  $\binom{[n]}{k}_{\text{stab}}$ , where two sets are adjacent if they are disjoint. We define the search problem SCHRIJVER as follows.

► **Definition 16** (Schrijver Graph Problem). *In the SCHRIJVER problem, the input consists of a Boolean circuit that represents a coloring*

$$c : \binom{[n]}{k}_{\text{stab}} \rightarrow [n - 2k + 1]$$

*of the Schrijver graph  $S(n, k)$  using  $n - 2k + 1$  colors, where  $n$  and  $k$  are integers satisfying  $n \geq 2k$ . The goal is to find a monochromatic edge, i.e., two disjoint sets  $S_1, S_2 \in \binom{[n]}{k}_{\text{stab}}$  such that  $c(S_1) = c(S_2)$ .*

As mentioned earlier, it was proved by Schrijver [40] that the chromatic number of  $S(n, k)$  is precisely  $n - 2k + 2$ . Therefore, every input to the SCHRIJVER problem has a solution.

#### 3.1 From Fair-IS-Cycle to Schrijver

The following theorem is used to obtain the hardness result for the SCHRIJVER problem. The proof applies an argument of [1] (see also [10]).

► **Theorem 17.** *The FAIR-IS-CYCLE problem is polynomial-time reducible to the SCHRIJVER problem.*

**Proof.** Consider an instance of the FAIR-IS-CYCLE problem, namely, a cycle  $G$  and a partition  $V_1, \dots, V_m$  of its vertex set into  $m$  sets. For every  $i \in [m]$ , let  $V'_i$  be the set obtained from  $V_i$  by removing one arbitrary vertex if  $|V_i|$  is even, and let  $V'_i = V_i$  otherwise. Since the size of every set  $V'_i$  is odd, we can write  $|V'_i| = 2r_i + 1$  for an integer  $r_i \geq 0$ . Let  $G'$  be the cycle obtained from  $G$  by removing the vertices that do not belong to the sets  $V'_i$  and connecting the remaining vertices according to their order in  $G$ . Letting  $n$  denote the number of vertices in  $G'$ , it can be assumed that its vertex set is  $[n]$  with the numbering from 1 to  $n$  along the cycle. Put  $k = \sum_{i=1}^m r_i$ , and notice that  $n = 2k + m$ . Define a coloring  $c$  of the Schrijver graph  $S(n, k)$  as follows. The color  $c(S)$  of a vertex  $S \in \binom{[n]}{k}_{\text{stab}}$  is defined as the smallest integer  $i \in [m]$  for which  $|S \cap V'_i| > r_i$  in case that such an  $i$  exists, and  $m + 1$  otherwise. This gives us a coloring of  $S(n, k)$  with  $n - 2k + 1$  colors, and thus an instance of the SCHRIJVER problem. It can be seen that a Boolean circuit that computes the coloring  $c$  can be constructed in polynomial running-time.

To prove the correctness of the reduction, consider a solution to the constructed SCHRIJVER instance, i.e., two disjoint sets  $S_1, S_2 \in \binom{[n]}{k}_{\text{stab}}$  with  $c(S_1) = c(S_2)$ . It is impossible that for some  $i \in [m]$  it holds that  $|S_1 \cap V'_i| > r_i$  and  $|S_2 \cap V'_i| > r_i$ , because  $S_1$  and  $S_2$  are disjoint and  $|V'_i| = 2r_i + 1$ . It follows that  $c(S_1) = c(S_2) = m + 1$ , meaning that  $|S_1 \cap V'_i| \leq r_i$  and  $|S_2 \cap V'_i| \leq r_i$  for all  $i \in [m]$ . Since  $|S_1| = |S_2| = k$ , it follows that



$|S_1 \cap V'_i| = r_i$  and  $|S_2 \cap V'_i| = r_i$  for all  $i \in [m]$ , hence  $S_1$  and  $S_2$  are two disjoint independent sets of  $G'$  covering all vertices but one from each  $V'_i$  and for each  $j \in \{1, 2\}$ , we have  $|S_j \cap V'_i| = \frac{1}{2} \cdot (|V'_i| - 1) \geq \frac{1}{2} \cdot |V_i| - 1$  for all  $i \in [m]$ . Since  $S_1$  and  $S_2$  are also independent sets of the original cycle  $G$ , each of them forms a valid solution to the FAIR-IS-CYCLE instance, completing the proof.  $\blacktriangleleft$

### 3.2 Membership in PPA

We now show that the SCHRIJVER and FAIR-SPLIT-CYCLE problems lie in PPA by reductions to the search problem associated with the Octahedral Tucker lemma. The reductions follow the proofs of the corresponding mathematical statements by Meunier [36] and by Alishahi and Meunier [3]. The proofs can be found in the full version of the paper.

We start with some notation (following [16, Section 2]). The partial order  $\preceq$  on the set  $\{+, -, 0\}$  is defined by  $0 \preceq +$  and by  $0 \preceq -$ , where  $+$  and  $-$  are incomparable. The definition is extended to vectors, so that for two vectors  $x, y$  in  $\{+, -, 0\}^n$ , we have  $x \preceq y$  if for all  $i \in [n]$  it holds that  $x_i \preceq y_i$  (equivalently,  $x_i = y_i$  whenever  $x_i \neq 0$ ). The Octahedral Tucker lemma, given implicitly in [33] and explicitly in [42], asserts that for every function  $\lambda : \{+, -, 0\}^n \setminus \{0\} \rightarrow \{\pm 1, \dots, \pm(n-1)\}$  satisfying  $\lambda(-x) = -\lambda(x)$  for all  $x$ , there exist vectors  $x, y$  such that  $x \preceq y$  and  $\lambda(x) = -\lambda(y)$ . This guarantees the existence of a solution to every input of the following search problem, denoted OCTAHEDRAL-TUCKER.

► **Definition 18** (Octahedral Tucker Problem). *In the OCTAHEDRAL-TUCKER problem, the input consists of a Boolean circuit that represents a function  $\lambda : \{+, -, 0\}^n \setminus \{0\} \rightarrow \{\pm 1, \pm 2, \dots, \pm(n-1)\}$  satisfying  $\lambda(-x) = -\lambda(x)$  for all  $x$ . The goal is to find vectors  $x, y$  such that  $x \preceq y$  and  $\lambda(x) = -\lambda(y)$ .*

The OCTAHEDRAL-TUCKER problem is known to be PPA-complete [17], where its membership in PPA follows already from [38] (see also [17, Appendix A]).

► **Proposition 19** ([38]). *The OCTAHEDRAL-TUCKER problem lies in PPA.*

The SCHRIJVER problem is reduced to OCTAHEDRAL-TUCKER, applying an argument of [36]. The proof is omitted.

► **Theorem 20.** *SCHRIJVER is polynomial-time reducible to OCTAHEDRAL-TUCKER.*

The FAIR-SPLIT-CYCLE problem (see Definition 4) is reduced to OCTAHEDRAL-TUCKER, applying an argument of [3]. The proof is omitted.

► **Theorem 21.** *FAIR-SPLIT-CYCLE is polynomial-time reducible to OCTAHEDRAL-TUCKER.*

### 3.3 Putting All Together

The presented reductions complete the proofs of our results. Indeed, the FAIR-IS-CYCLE problem is PPA-hard by Corollary 14, and is polynomial-time reducible to the FAIR-SPLIT-CYCLE and SCHRIJVER problems by Lemma 15 and Theorem 17 respectively. By Theorems 20 and 21, each of the two is efficiently reducible to the OCTAHEDRAL-TUCKER problem, which by Proposition 19 lies in PPA. It thus follows that all of these problems are PPA-complete (see Figure 1), confirming Theorems 3, 5, and 7.



## References

- 1 Ron Aharoni, Noga Alon, Eli Berger, Maria Chudnovsky, Dani Kotlar, Martin Loebl, and Ran Ziv. Fair representation by independent sets. In M. Loebl, J. Nešetřil, and R. Thomas, editors, *A Journey Through Discrete Mathematics: A Tribute to Jiří Matoušek*, pages 31–58. Springer, 2017.
- 2 James Aisenberg, Maria Luisa Bonet, and Sam Buss. 2-D Tucker is PPA complete. *J. Comput. Syst. Sci.*, 108:92–103, 2020.
- 3 Meysam Alishahi and Frédéric Meunier. Fair splitting of colored paths. *Electron. J. Comb.*, 24(3):P3.41, 2017.
- 4 Noga Alon. Combinatorial Nullstellensatz. *Combinatorics, Probability and Computing*, 8(1–2):7–29, 1999.
- 5 Noga Alon. Discrete mathematics: methods and challenges. In *Proc. of the International Congress of Mathematicians (ICM’02)*, pages 119–135. Higher Education Press, 2002.
- 6 Noga Alon and Michael Tarsi. Colorings and orientations of graphs. *Combinatorica*, 12(2):125–134, 1992.
- 7 Noga Alon and Douglas B. West. The Borsuk-Ulam theorem and bisection of necklaces. *Proc. Amer. Math. Soc.*, 98(4):623–628, 1986.
- 8 Aleksandrs Belovs, Gábor Ivanyos, Youming Qiao, Miklos Santha, and Siyi Yang. On the polynomial parity argument complexity of the combinatorial nullstellensatz. In *32nd Computational Complexity Conference (CCC’17)*, pages 30:1–30:24, 2017.
- 9 Kristóf Bérczi and Yusuke Kobayashi. An algorithm for identifying cycle-plus-triangles graphs. *Discret. Appl. Math.*, 226:10–16, 2017.
- 10 Alexander Black, Umur Cetin, Florian Frick, Alexander Pacun, and Linus Setiabrata. Fair splittings by independent sets in sparse graphs. *Israel J. of Math.*, 236:603–627, 2020.
- 11 Karol Borsuk. Drei Sätze über die  $n$ -dimensionale euklidische sphäre. *Fundamenta Mathematicae*, 20(1):177–190, 1933.
- 12 Xi Chen and Xiaotie Deng. On the complexity of 2D discrete fixed point problem. *Theor. Comput. Sci.*, 410(44):4448–4456, 2009. Preliminary version in ICALP’06.
- 13 Xi Chen, Xiaotie Deng, and Shang-Hua Teng. Settling the complexity of computing two-player Nash equilibria. *J. ACM*, 56(3):14:1–14:57, 2009. Preliminary version in FOCS’06.
- 14 Constantinos Daskalakis, Paul W. Goldberg, and Christos H. Papadimitriou. The complexity of computing a Nash equilibrium. *SIAM J. Comput.*, 39(1):195–259, 2009. Preliminary version in STOC’06.
- 15 Constantinos Daskalakis and Christos H. Papadimitriou. Continuous local search. In *Proc. of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA’11)*, pages 790–804, 2011.
- 16 Jesús A. De Loera, Xavier Goaoc, Frédéric Meunier, and Nabil H. Mustafa. The discrete yet ubiquitous theorems of Carathéodory, Helly, Sperner, Tucker, and Tverberg. *Bull. Amer. Math. Soc.*, 56(3):415–511, 2019.
- 17 Xiaotie Deng, Zhe Feng, and Rucha Kulkarni. Octahedral Tucker is PPA-complete. *Electronic Colloquium on Computational Complexity (ECCC)*, 24:118, 2017.
- 18 Xiaotie Deng, Qi Qi, and Amin Saberi. Algorithmic solutions for envy-free cake cutting. *Oper. Res.*, 60(6):1461–1476, 2012.
- 19 D. Z. Du, D. F. Hsu, and F. K. Hwang. The Hamiltonian property of consecutive- $d$  digraphs. *Math. and Computer Modelling*, 17(11):61–63, 1993.
- 20 Paul Erdős. On some of my favourite problems in graph theory and block designs. *Matematiche*, 45(1):61–73, 1990.
- 21 John Fearnley, Spencer Gordon, Ruta Mehta, and Rahul Savani. Unique end of potential line. In *46th International Colloquium on Automata, Languages, and Programming (ICALP’19)*, pages 56:1–56:15, 2019.

- 22 Aris Filos-Ratsikas, Søren Kristoffer Stiil Frederiksen, Paul W. Goldberg, and Jie Zhang. Hardness results for consensus-halving. In *43rd International Symposium on Mathematical Foundations of Computer Science (MFCS'18)*, pages 24:1–24:16, 2018.
- 23 Aris Filos-Ratsikas and Paul W. Goldberg. Consensus halving is PPA-complete. In *Proc. of the 50th Annual ACM SIGACT Symposium on Theory of Computing (STOC'18)*, pages 51–64, 2018.
- 24 Aris Filos-Ratsikas and Paul W. Goldberg. The complexity of splitting necklaces and bisecting ham sandwiches. In *Proc. of the 51st Annual ACM SIGACT Symposium on Theory of Computing (STOC'19)*, pages 638–649, 2019.
- 25 Aris Filos-Ratsikas, Alexandros Hollender, Katerina Sotiraki, and Manolis Zampetakis. Consensus-halving: Does it ever get easier? In *21st ACM Conference on Economics and Computation (EC'20)*, pages 381–399, 2020.
- 26 Herbert Fleischner and Michael Stiebitz. A solution to a colouring problem of P. Erdős. *Discret. Math.*, 101(1–3):39–48, 1992.
- 27 Herbert Fleischner and Michael Stiebitz. Some remarks on the cycle plus triangles problem. In *The Mathematics of Paul Erdős II*, volume 14, pages 136–142. Springer, 1997.
- 28 Charles H. Goldberg and Douglas B. West. Bisection of circle colorings. *SIAM J. Alg. Disc. Meth.*, 6(1):93–106, 1985.
- 29 Paul W. Goldberg and Alexandros Hollender. The hairy ball problem is PPAD-complete. In *46th International Colloquium on Automata, Languages, and Programming (ICALP'19)*, pages 65:1–65:14, 2019.
- 30 David S. Johnson, Christos H. Papadimitriou, and Mihalis Yannakakis. How easy is local search? *J. Comput. Syst. Sci.*, 37(1):79–100, 1988. Preliminary version in FOCS'85.
- 31 Martin Kneser. Aufgabe 360. *Jahresbericht der Deutschen Mathematiker-Vereinigung*, 58(2):27, 1955.
- 32 László Lovász. Kneser's conjecture, chromatic number, and homotopy. *J. Comb. Theory, Ser. A*, 25(3):319–324, 1978.
- 33 Jiří Matoušek. A combinatorial proof of Kneser's conjecture. *Combinatorica*, 24(1):163–170, 2004.
- 34 Jiří Matoušek. *Using the Borsuk-Ulam Theorem: Lectures on Topological Methods in Combinatorics and Geometry*. Springer Publishing Company, Incorporated, 2007.
- 35 Nimrod Megiddo and Christos H. Papadimitriou. On total functions, existence theorems and computational complexity. *Theor. Comput. Sci.*, 81(2):317–324, 1991.
- 36 Frédéric Meunier. The chromatic number of almost stable Kneser hypergraphs. *J. Comb. Theory, Ser. A*, 118(6):1820–1828, 2011.
- 37 Dömötör Pálvölgyi. 2D-Tucker is PPAD-complete. In *5th International Workshop on Internet and Network Economics (WINE'09)*, pages 569–574, 2009.
- 38 Christos H. Papadimitriou. On the complexity of the parity argument and other inefficient proofs of existence. *J. Comput. Syst. Sci.*, 48(3):498–532, 1994.
- 39 Horst Sachs. Elementary proof of the cycle-plus-triangles theorem. In *Combinatorics, Paul Erdős is eighty*, volume 1, pages 347–359. Bolyai Soc. Math. Stud., 1993.
- 40 Alexander Schrijver. Vertex-critical subgraphs of Kneser graphs. *Nieuw Arch. Wiskd.*, 26(3):454–461, 1978.
- 41 Forest W. Simmons and Francis Edward Su. Consensus-halving via theorems of Borsuk-Ulam and Tucker. *Math. Soc. Sci.*, 45(1):15–25, 2003.
- 42 Günter M. Ziegler. Generalized Kneser coloring theorems with combinatorial proofs. *Invent. math.*, 147(3):671–691, 2002.

# Sharp Threshold Rates for Random Codes

**Venkatesan Guruswami**

Computer Science Department, Carnegie Mellon University, Pittsburgh, PA, USA  
venkatg@cs.cmu.edu

**Jonathan Mosheiff**

Computer Science Department, Carnegie Mellon University, Pittsburgh, PA, USA  
jmosheiff@cs.cmu.edu

**Nicolas Resch**

Centrum Wiskunde & Informatica, Amsterdam, The Netherlands  
nresch@cs.cmu.edu

**Shashwat Silas**

Computer Science Department, Stanford University, CA, USA  
silas@stanford.edu

**Mary Wootters**

Computer Science Department and Electrical Engineering Department,  
Stanford University, CA, USA  
marykw@stanford.edu

---

## Abstract

Suppose that  $\mathcal{P}$  is a property that may be satisfied by a random code  $C \subset \Sigma^n$ . For example, for some  $p \in (0, 1)$ ,  $\mathcal{P}$  might be the property that there exist three elements of  $C$  that lie in some Hamming ball of radius  $pn$ . We say that  $R^*$  is the *threshold rate* for  $\mathcal{P}$  if a random code of rate  $R^* + \varepsilon$  is very likely to satisfy  $\mathcal{P}$ , while a random code of rate  $R^* - \varepsilon$  is very unlikely to satisfy  $\mathcal{P}$ . While random codes are well-studied in coding theory, even the threshold rates for relatively simple properties like the one above are not well understood.

We characterize threshold rates for a rich class of properties. These properties, like the example above, are defined by the inclusion of specific sets of codewords which are also suitably “symmetric.” For properties in this class, we show that the threshold rate is in fact *equal* to the lower bound that a simple first-moment calculation obtains. Our techniques not only pin down the threshold rate for the property  $\mathcal{P}$  above, they give sharp bounds on the threshold rate for *list-recovery* in several parameter regimes, as well as an efficient algorithm for estimating the threshold rates for list-recovery in general.

**2012 ACM Subject Classification** Mathematics of computing  $\rightarrow$  Coding theory

**Keywords and phrases** Coding theory, Random codes, Sharp thresholds

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.5

**Related Version** <https://arxiv.org/abs/2009.04553>

**Funding** *Venkatesan Guruswami*: Supported by NSF grants CCF-1563742 and CCF-1814603 and a Simons Investigator Award.

*Jonathan Mosheiff*: Supported by NSF grants CCF-1563742 and CCF-1814603 and a Simons Investigator Award.

*Nicolas Resch*: Supported by NSF grants CCF-1563742 and CCF-1814603, ERC H2020 grant No.74079 (ALGSTRONGCRYPTO), and a Simons Investigator Award.

*Shashwat Silas*: Supported by NSF-CAREER grant CCF-1844628, NSF-BSF grant CCF-1814629, a Sloan Research Fellowship, and a Google Graduate Fellowship.

*Mary Wootters*: Supported by NSF-CAREER grant CCF-1844628, NSF-BSF grant CCF-1814629, and a Sloan Research Fellowship.

**Acknowledgements** We would like to thank Ray Li for helpful conversations.



© Venkatesan Guruswami, Jonathan Mosheiff, Nicolas Resch, Shashwat Silas, and Mary Wootters; licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 5; pp. 5:1–5:20



Leibniz International Proceedings in Informatics  
LIPIcs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

## 1 Introduction

Random codes are ubiquitous in the theory of error correcting codes: when thinking about the “right” trade-offs for a particular problem, a coding theorist’s first instinct may be to try a random code. A *random code* here is simply a random set. That is, let  $C \subseteq \Sigma^n$  be chosen so that each  $x \in \Sigma^n$  is included in  $C$  with probability  $|\Sigma|^{-n(1-R)}$  for some parameter  $R$ , which is called the (expected<sup>1</sup>) *rate* of the code  $C$ . Random codes are used in the proofs of the Gilbert-Varshamov bound, Shannon’s channel coding theorem, and the list-decoding capacity theorem, to name just a few. This success may lead to the intuition that random codes are “easy” to analyze, and that the hard part is finding explicit constructions that match (or in rare cases, exceed) the parameters of random codes. However, there is still much we do not know about random codes, especially if we want extremely precise answers.

In particular, the question of *threshold rates*, of broader interest in probability theory, is something that we do not understand well for random codes. In more detail, suppose that  $\mathcal{P}$  is a code property. For example, perhaps  $\mathcal{P}$  is the property that there is some pair of codewords  $c^{(1)}, c^{(2)} \in C$  that both lie in some Hamming ball of radius  $pn$ . Or perhaps  $\mathcal{P}$  is the property that there are three codewords  $c^{(1)}, c^{(2)}, c^{(3)} \in C$  that lie in such a Hamming ball. A value  $R^* \in (0, 1)$  is a *threshold rate* for  $\mathcal{P}$  if a random code of rate  $R^* + \varepsilon$  is very likely to satisfy  $\mathcal{P}$ , but a random code of rate  $R^* - \varepsilon$  is very unlikely to satisfy  $\mathcal{C}$ . For the first example above, about pairs of codewords, the property in question is just the property of the code having *minimum distance* less than  $2pn$ , and this is not too hard to understand. However, already for the second example above – called *list-of-two decoding* – the threshold rate was not known.

### 1.1 Contributions

In this paper, we characterize threshold rates for a rich class of natural properties of random codes. We apply our characterization to obtain threshold rates for list-of-two decoding, as well as to properties like *list-decoding* and *perfect hashing codes*, and more generally to *list-recovery*. We outline our contributions below.

#### A characterization of the threshold rate $R^*$ for symmetric properties

Suppose that  $\mathcal{P}$  is a property defined by the inclusion of certain “bad” sets. For example, the list-of-two decoding property described above is defined by the inclusion of three codewords that lie in a radius- $pn$  Hamming ball. For such properties that are also “symmetric enough,” our main technical result, Theorem 1, characterizes the threshold rate  $R^*$ . Moreover, we show that this threshold rate is exactly the same as the lower bound that one obtains from a simple first-moment calculation! This is in contrast to recent work of [13] for random *linear* codes, which shows that the corresponding first-moment calculation is not the correct answer in that setting.

Part of our contribution is formalizing the correct notion of “symmetric enough.” As we describe in the technical overview in Section 1.2, this definition turns out to be fairly subtle. We also show in the full version of the paper, that this definition is necessary.

---

<sup>1</sup> Throughout, we refer to  $R$  as the rate of the code, and drop the adjective “expected.”

### Estimates of $R^*$ for list-recovery

We give precise estimates of the threshold rate  $R^*$  for *list-recovery*. We say that a code  $C \subseteq \Sigma^n$  is  $(p, \ell, L)$ -list-recoverable if for all sets  $K_i \subseteq \Sigma$  (for  $1 \leq i \leq n$ ) with  $|K_i| \leq \ell$ ,

$$|\{c \in C : \Pr_{i \sim [n]}[c_i \notin K_i] \leq p\}| < L.$$

List-recovery is a useful primitive in list-decoding, algorithm design, and pseudorandomness (see, e.g., [15, 10, 16]). In particular, it generalizes the list-of-two decoding example above (when  $\ell = 1$  and  $L = 3$ ), as well as other interesting properties, such as list-decoding and perfect hashing codes, discussed below.

Our characterization allows us to estimate or even exactly compute the threshold rate for  $(p, \ell, L)$ -list-recovery in a wide variety of parameter regimes. To demonstrate this, we include several results along these lines. First, in Section 4 (Corollary 38), we give estimates that are quite sharp when  $\frac{q \log L}{L}$  is small. In Section 5 (Lemma 40), we give an exact formula for the case  $p = 0$ , which is relevant for perfect hashing codes. In Section 6 (Theorem 42(I)), we give an exact formula for the case that  $L = 3$  and  $\ell = 1$ , relevant for list-of-two decoding. Moreover, in Section 7 (Corollary 47) we use our characterization to develop an efficient algorithm to compute the threshold rate up to an additive error of  $\varepsilon > 0$ ; our algorithm runs in time  $O_p(L^q + \text{poly}(q, L, \log(1/\varepsilon)))$ .

### List-of-two decoding and a separation between random codes and random linear codes

We obtain new results for list-of-two decoding, the example discussed above. List-of-two decoding is a special case of *list-decoding*, which itself the special case of list-recovery where  $\ell = 1$ . We say that a code is  $(p, L)$ -list-decodable if there is no Hamming ball of radius  $pn$  containing  $L$  codewords; list-of-two decoding is the special case of  $L = 3$ .<sup>2</sup> We show in Section 6 (Theorem 42) that the threshold rate for this question, for random binary codes, is  $R^* = 1 - \frac{1-h_2(3p)+3p \log_2 3}{3}$ . That is, above this rate, a random binary code is very likely to have three codewords contained in a radius  $pn$  ball, while below this rate, the code most likely avoids all such triples.

This result is interesting for two reasons. First, it demonstrates that our techniques are refined enough to pin down the threshold rate in this parameter regime. Second, the particular value of  $R^*$  is interesting because it is *different* than the corresponding threshold rate for random *linear* codes. A *random linear code* over  $\mathbb{F}_q$  of rate  $R$  is a random linear subspace of  $\mathbb{F}_q^n$ , of dimension  $Rn$ . The list-decodability of random linear codes has been extensively studied, and it is known (e.g., [19, 7]) that the  $(p, L)$ -list-decoding threshold rate for both random linear codes and random codes is  $1 - h_q(p)$ , for sufficiently large list sizes  $L$ .<sup>3</sup>

### Limitations of random codes for perfect hashing

Another special case of list-recovery is *perfect hashing codes*. Suppose that  $|\Sigma| = q$ . A code  $C \subseteq \Sigma^n$  is said to be a  $q$ -hash code if, for any set of  $q$  distinct codewords  $c^{(1)}, c^{(2)}, \dots, c^{(q)} \in C$ , there is at least one  $i \in [n]$  so that  $\{c_i^{(1)}, c_i^{(2)}, \dots, c_i^{(q)}\} = \Sigma$ ; that is, if the set of symbols that

<sup>2</sup> It is called list-of-two decoding, even though  $L$  is *three*, because any Hamming ball contains at most *two* codewords.

<sup>3</sup> Here,  $h_q(x) = x \log_q(q-1) - x \log_q(x) - (1-x) \log_q(1-x)$  is the  $q$ -ary entropy.

appear in position  $i$  are all distinct. Thus,  $C$  is a  $q$ -hash code if and only if it is  $(0, q-1, q)$ -list-recoverable. As the name suggests,  $q$ -hash codes have applications in constructing small perfect hash families, and it is a classical question to determine the largest rate possible for a  $q$ -hash code.<sup>4</sup>

A simple random coding argument shows that a random code of rate  $R = \frac{1}{q} \log_q \frac{1}{1-q^{1/q^q}} - o(1)$  is a  $q$ -hash code with high probability [5, 12]. However, it is still an area of active research to do significantly better than this bound for any  $q$ . It is known that  $R < \frac{q^l}{q^q-1}$  for any  $q$ -hash code [5, 9], and for large  $q$ , there is a gap of a multiplicative factor of about  $q^2$  between these upper and lower bounds. Körner and Matron gave a construction that beats the random bound for  $q = 3$  [11], and recently Xing and Yuan gave a construction that beats the random bound for infinitely many  $q$ 's [18]. One might have hoped that a random code might in fact do better than the straightforward probabilistic argument (which follows from a union bound). Unfortunately, our results show that this is not the case.

### A broader view

Taking a broader view, threshold phenomena in other combinatorial domains, notably random graphs and Boolean functions, have been the subject of extensive study at least since Erdős and Rényi's seminal work [3]. Some of the deeper results in this field (e.g. [6]), deal simultaneously with a wide class of properties, rather than a specific one. Other works, such as the recent [4], are general enough to cover not only multiple properties, but also multiple domains. Our work (as with the work of [13], [8] on random linear codes, discussed below) is not as general as these, but we are able to get more precise results. It would be interesting to find a general framework that connects threshold phenomena in a variety of random code models, with analogues from random graphs and other natural combinatorial structures.

## 1.2 Technical Overview

As mentioned above, we study properties defined by the inclusion of bad subsets. We organize bad subsets of size  $b$  into matrices  $B \in \Sigma^{n \times b}$ , interpreting the columns of  $B$  as the elements of the set. We write " $B \subseteq C$ " to mean that the columns of  $B$  are all contained in the code  $C$ .

As a running example – and also our motivating example – consider list recovery, defined above. The property  $\mathcal{P}$  of *not* being  $(p, \ell, L)$ -list-recoverable is defined by the inclusion of "bad" matrices  $B \in \Sigma^{n \times L}$  so that for some sets  $K_1, \dots, K_n \subset \Sigma$  of size at most  $\ell$ ,  $\Pr_{i \sim [n]}[B_{ij} \notin K_i] \leq p$  for each  $j \in [L]$ . Moreover we require the columns of  $B$  to be distinct.

### Analyzing a property as a union of types

Following the approach of [13] for random linear codes, we group the bad matrices into *types* based on their row distributions. That is, for a bad matrix  $B \in \Sigma^{n \times b}$ , let  $\tau$  denote the row distribution

$$\tau(v) = \frac{|\{i \in [n] : B_{i,\star} = v\}|}{n},$$

where  $B_{i,\star}$  denotes the  $i$ 'th row of  $B$ . We say that  $B$  has *type*  $\tau$ . Consider the set  $\mathcal{B}$  of all of the matrices of type  $\tau$ ; equivalently,  $\mathcal{B}$  is the set of matrices obtained by permuting the

<sup>4</sup> A  $q$ -hash code naturally gives rise to a perfect hash family: suppose that  $C$  is a universe of items, and define a hash function  $h_i : C \rightarrow \Sigma$  given by  $h_i(c) = c_i$ . Then the property of being a  $q$ -hash code is equivalent to the property that, for any set of  $q$  items in the universe, there exists some hash function  $h_i$  for  $1 \leq i \leq n$  that maps each item to a different value.



rows of  $B$ . We note that possible types  $\tau$  depend on  $n$ , because of divisibility constraints. For simplicity, let us ignore these restrictions for now (we will deal with them later), and suppose that a single type  $\tau$  can appear for all  $n$ .

### First-moment bound and main theorem

We can use a simple first-moment approach to give a lower bound on the threshold rate. In more detail, the probability that a particular  $B$  is contained in  $C$  is  $q^{-nb(1-R)}$ , assuming that  $B$  has  $b$  distinct columns. Using the fact that  $|\mathcal{B}| \approx q^{H_q(\tau) \cdot n}$ , where  $H_q(\tau)$  is the base- $q$  entropy of  $\tau$  (see Section 2), and applying a union bound over all  $B \in \mathcal{B}$ , we see that the probability that any  $B \in \mathcal{B}$  is contained in  $C$  is at most

$$q^{nb(H_q(\tau)-(1-R))}.$$

Thus, if  $R \leq 1 - \frac{H_q(\tau)}{b} - \varepsilon$  for some small  $\varepsilon > 0$ , it is very unlikely that  $\tau$  will be represented in  $C$ .

Now suppose that our collection of bad sets, which define the property  $\mathcal{P}$ , is closed under row permutations. This means that  $\mathcal{P}$  can be represented as a collection  $T$  of types  $\tau$ ; note that the size of  $T$  is polynomial in  $n$ . Union bounding over all of these types, the computation above shows that a random code  $C$  of rate  $R < 1 - \max_{\tau \in T} \frac{H_q(\tau)}{b} - \varepsilon$  will, with high probability, not satisfy  $\mathcal{P}$ .

The question is, could the rate be larger? Might it be the case that  $\mathcal{P}$  still not satisfied (with high probability) by a random code of rate  $R$  significantly larger than  $1 - \max_{\tau} H_q(\tau)/b$ ? In [13], it was shown that the answer for random *linear* codes is “yes.” If  $\mathcal{P}$  exhibits certain linear structure, then it may be possible that a higher rate random linear code still does not satisfy  $\mathcal{P}$  with high probability. One may conjecture that something similar holds for random codes.

Our main technical result, Theorem 30, is that, for random codes, for sufficiently symmetric properties, the answer to this question is “no.” That is, the simple calculation above *does* give the right answer for random codes!

► **Theorem 1** (Informal; see Theorem 30 for the formal version). *Let  $\mathcal{P}$  be a “symmetric” property defined by the inclusion of a type among the types in  $T$ . Let*

$$R^* = 1 - \frac{\max_{\tau \in T} H_q(\tau)}{b}$$

*Then for all  $\varepsilon > 0$ , a random code of rate  $R \geq R^* + \varepsilon$  satisfies  $\mathcal{P}$  with probability  $1 - o(1)$ , while a random code of rate  $R^* - \varepsilon$  satisfies  $\mathcal{P}$  with probability  $o(1)$ .*

### Sketch of proof: second moment method

Below, we sketch the proof of Theorem 1, and explain what the assumption of “symmetry” means. As noted above, it is straightforward to show that the threshold rate  $R^*$  is at least  $1 - \max_{\tau \in T} \frac{H_q(\tau)}{b}$ , so the challenge is to show that it is not larger. The proof of Theorem 1 uses the second-moment method to show that for any *histogram type*  $\tau$  (we discuss histogram types more below), a random code  $C$  of rate  $1 - H_q(\tau)/b + \varepsilon$  is very likely to contain some matrix  $B$  with type  $\tau$ . Thus, the threshold rate is at most  $1 - \max_{\tau} H_q(\tau)/b$ , where the maximum is over all histogram types  $\tau$  that appear in  $T$ . Our eventual definition of “symmetric” will guarantee that it is legitimate to restrict our attention to histogram types.

### Histogram types and the meaning of “symmetry”

In order to apply the second moment method, we bound the variance of  $\sum_{B \sim \tau} \mathbf{1}[B \subset C]$ , where the sum is over all matrices  $B$  of type  $\tau$ . This turns out to be possible when  $\tau$  has the following symmetry property: for any  $u \in \Sigma^b$ , and for any permutation  $\pi : [b] \rightarrow [b]$ , it holds that  $\tau(u) = \tau(\pi(u))$ , where  $\pi(u)$  denotes the corresponding coordinate permutation of  $u$ . We call such a type  $\tau$  a *histogram-type* (Definition 27) because the probability of a particular vector  $u$  under  $\tau$  depends only on the histogram of  $u$ .

A first attempt to formulate a definition of “symmetry” for Theorem 1 is thus to require  $\mathcal{P}$  to be defined by histogram types. This results in a true statement, but unfortunately it is too restrictive: it is not hard to see that, for example, the property of not being list-decodable contains types  $\tau$  that are not histogram types. Fortunately, for the logic above to go through, it is enough to show that  $T$  contains a type  $\tau$  that is *both* a maximum entropy distribution in  $T$ , and is also a histogram type. Thus, the assumption of “symmetry” we will use is that  $T$ , the collection of types represented in the property  $\mathcal{P}$ , forms a convex set. Then, using the fact that  $\mathcal{P}$  is defined by the inclusion of bad sets (which do not care about the order of the columns in the corresponding matrices), we can always find a maximum entropy histogram type by “symmetrizing” and taking a convex combination of column permutations of some maximum entropy type  $\tau$ . One might wonder if this symmetrization step (and the resulting assumption about convexity) is necessary. In the full version of this paper show that it is.

### Taking a limit as $n \rightarrow \infty$

There is one more challenge to consider, which is that in the description above, we have ignored the fact that we would like our characterization to work for a sequence of values of  $n$ . However, a type  $\tau$  only works for certain values of  $n$  due to divisibility restrictions. To get around this, we work instead with a sequence of types  $\tau_n$  which tend to  $\tau$ . This leads us to our final definition of “symmetric” (Definition 20). Suppose that  $\mathcal{P}$  is a property defined by the inclusion of size- $b$  bad sets. Then for each  $n$ , there is some collection  $T^n$  of bad types  $\tau_n$ , each of which is a distribution on  $\Sigma^b$ . We say that  $\mathcal{P}$  is *symmetric* if the sets  $T^n$  approach some convex set  $T$  as  $n$  goes to infinity. The logic above then goes through to give Theorem 1.

### Applications to list-recovery

Finally, in order to apply Theorem 1, we need to understand the maximum entropy distribution  $\tau$  for our property  $\mathcal{P}$ . We do this for the property  $\mathcal{P}$  of not being  $(p, \ell, L)$ -list-recoverable in a variety of parameter regimes in Sections 4, 5 and 6, and along the way obtain our results about list-of-two decoding and perfect hashing codes. Finally, in Section 7, we use our framework to develop an algorithm to efficiently calculate the threshold rate for  $(p, \ell, L)$ -list-recovery.

## 1.3 Organization

In Section 2, we introduce notation, and also set up the definitions we need about types, thresholds, properties, and various notions of symmetry. We also introduce (non-)list-recoverability as a property, and prove in Corollary 24 that it is symmetric.

In Section 3, we state and prove Theorem 30, the formal version of the characterization theorem (Theorem 1 above). At the end of Section 3, we begin to apply Theorem 30 to list-recovery, and in particular define several notions we will need to analyze list recovery in the subsequent sections.



In the remaining sections, we specialize to list-recovery. Note that the proofs of the claims in the remaining sections are available in the full version. In Section 4, we develop bounds on the threshold rate  $R^*$  for list-recovery that are tight when  $(q \log L)/L$  is small. In Section 5, we compute the threshold rate  $R^*$  exactly for zero-error list-recovery (that is, when  $p = 0$ ), and use this to compute the threshold rate for perfect hashing. In Section 6, we compute the threshold rate  $R^*$  for list-of-two decoding (e.g., list-recovery when  $\ell = 1$  and  $L = 3$ ), and use this to quantify the gap between random codes and random linear codes for list-of-two decoding. Finally, in Section 7, we give an efficient algorithm to compute the threshold rate.

## 2 Preliminaries

First, we fix some basic notation. Throughout, we consider codes  $C \subseteq \Sigma^n$  of block length  $n$  over an alphabet  $\Sigma$ , where  $|\Sigma| = q$ . When we use  $\log(x)$  without an explicit base, we mean  $\log_2(x)$ . We use  $H_q$  to denote the base- $q$  entropy: for a distribution  $\tau$ ,

$$H_q(\tau) := - \sum_x \tau(x) \log_q(\tau(x)).$$

When  $q$  is clear from context, we will use  $H(\tau)$  to denote  $H_q(\tau)$ . If  $u$  is a random variable distributed according to  $\tau$ , then we abuse notation slightly and define  $H(u) := H(\tau)$ . We use  $h_q(x) := x \log_q(q-1) - x \log_q(x) - (1-x) \log_q(1-x)$  to denote the  $q$ -ary entropy of  $x \in (0, 1)$ . Again, when  $q$  is clear from context we will use  $h(x)$  to denote  $h_q(x)$ .

For a vector  $x \in \Sigma^k$  and  $I \subseteq [k]$ , we use  $x_I$  to refer to the vector  $(x_i)_{i \in I} \in \Sigma^I$ . Given a vector  $u \in \Sigma^k$  and a permutation  $\pi : [k] \rightarrow [k]$ , we let  $\pi(u) \in \Sigma^k$  denote the corresponding coordinate permutation of  $u$ .

Given distributions  $\tau, \mu$  on the same finite set, we define their  $\ell_\infty$ -distance by

$$d_\infty(\tau, \mu) := \max_x |\tau(x) - \mu(x)|.$$

Given a set of distributions  $T$ , we define the  $\ell_\infty$  distance from  $\mu$  to  $T$  by

$$d_\infty(\mu, T) := \inf_{\tau \in T} d_\infty(\mu, \tau).$$

### 2.1 Basic notions

As mentioned in the introduction, we will organize our “bad” sets into matrices. We formalize this with the following two definitions.

► **Definition 2** (Matrices with distinct columns). Let  $\Sigma_{\text{distinct}}^{n \times b}$  denote the collection of all matrices  $B \in \Sigma^{n \times b}$  such that each column of  $B$  is distinct.

► **Definition 3** (Subsets as matrices). Let  $C \subseteq \Sigma^n$  be a code, and let  $B \in \Sigma^{n \times b}$  be a matrix. We write  $B \subseteq C$  to mean that each column of  $B$  is an element of  $C$ . If  $A \subseteq \Sigma^n$ , let  $\mathcal{B}_A \subseteq \Sigma^{n \times |A|}$  denote the collection all matrices  $B \in \Sigma_{\text{distinct}}^{n \times |A|}$  such that the columns of  $B$  are the elements of  $A$ .

For completeness, we reiterate our definition of a random code from the introduction.

► **Definition 4** (Random code). Let  $\Sigma$  be a finite set with  $q := |\Sigma| \geq 2$ . For  $n \in \mathbb{N}$  and  $R \in [0, 1]$ , let  $C_{\text{RC}}^n(R)$  denote an expected-rate  $R$  random code (over the alphabet  $\Sigma$ )  $C \subseteq \Sigma^n$ . Namely, for each  $x \in \mathbb{F}_q^n$  we have  $\Pr[x \in C] = q^{-n(1-R)}$ , and these events are independent over all  $x$ .

## 5:8 Sharp Threshold Rates for Random Codes

We record a useful fact about random codes, which is the probability that any particular matrix  $B$  is contained in one.

► **Fact 5** (Probability that a random code contains a matrix). *Let  $B \in \Sigma^{n \times b}$ . Then,*

$$\Pr[B \subseteq C_{\text{RC}}^n(R)] = q^{-n(1-R)t},$$

where  $t$  is the number of distinct columns in  $B$ .

We study (noisy) list-recovery, which generalizes both the list-decoding and perfect hashing examples mentioned in the introduction. We repeat the definition, so that we may formally define a “bad” matrix for list-recovery.

► **Definition 6** (Noisy list-recovery). *Let  $p \in [0, 1]$ ,  $1 \leq \ell \leq q$ , and  $L \in \mathbb{N}$ . Say that a matrix  $B \in \Sigma_{\text{distinct}}^{L \times n}$  is  $(p, \ell, L)$ -bad for  $(p, \ell, L)$ -list-recovery if there exist sets  $K_i \subseteq \Sigma$  ( $1 \leq i \leq n$ ), each of size  $\ell$ , such that for every  $1 \leq j \leq L$ ,*

$$\Pr_{i \sim [n]} [B_{i,j} \notin K_i] \leq p.$$

*A code  $C \subseteq \Sigma^n$  is  $(p, \ell, L)$ -list-recoverable if it does not contain a  $(p, \ell, L)$ -bad matrix.*

### 2.2 Monotone-increasing properties and thresholds

We study the threshold rate  $R^*$  for random codes to satisfy certain properties. This was discussed informally in the introduction and the definitions below formalize what “threshold rate” means.

► **Definition 7** (Monotone-increasing property). *A code property  $\mathcal{P}$  is monotone-increasing if given a code  $C$  satisfying  $\mathcal{P}$ , it holds that every code  $C'$  such that  $C \subseteq C'$  also satisfies  $\mathcal{P}$ .*

For example, the property of being *not*  $(p, \ell, L)$ -list-recoverable (that is, the property of containing a  $(p, \ell, L)$ -bad matrix) is a monotone-increasing property.

► **Definition 8** (Minimal-set). *Let  $P_n$  be a monotone-increasing property of length- $n$  codes. A set  $A \subseteq \Sigma^n$  is a minimal element of  $P_n$  if  $A$  satisfies  $P_n$  but no strict subset of  $A$  satisfies  $P_n$ . The minimal set for  $P_n$  is the collection of matrices*

$$\bigcup_{A \text{ is a minimal element of } P_n} \mathcal{B}_A.$$

For example, the minimal set for the property  $P_n$  of being *not*  $(p, \ell, L)$ -list-recoverable is the set of  $(p, \ell, L)$ -bad matrices.

Note that a code satisfies  $P_n$  if and only if it contains some matrix belonging to the minimal set of  $P_n$ . If  $\mathcal{P}$  is a monotone-increasing property of codes, we define its associated *threshold rate* by

$$R_{\text{RC}}^n(\mathcal{P}) := \begin{cases} \sup \{R \in [0, 1] : \Pr[C_{\text{RC}}^n(R) \text{ satisfies } \mathcal{P}] \leq \frac{1}{2}\} & \text{if there is such an } R \\ 0 & \text{otherwise.} \end{cases}$$

► **Remark 9.** If  $\mathcal{P}$  is monotone-increasing then the function  $\Pr[C_{\text{RC}}^n(R) \text{ satisfies } \mathcal{P}]$  is monotone-increasing in  $R$ . This can be proved by a standard coupling argument, akin to [1, Thm. 2.1].

► **Definition 10** (Sharpness for random codes). *A monotone-increasing property  $\mathcal{P}$  is sharp for random codes if*

$$\lim_{n \rightarrow \infty} \Pr[C_{\text{RC}}^n(R_{\text{RC}}^n(\mathcal{P}) - \varepsilon) \text{ satisfies } \mathcal{P}] = 0$$

and

$$\lim_{n \rightarrow \infty} \Pr[C_{\text{RC}}^n(R_{\text{RC}}^n(\mathcal{P}) + \varepsilon) \text{ satisfies } \mathcal{P}] = 1$$

for every  $\varepsilon > 0$ .

### 2.3 Local and row-symmetric properties

As discussed in the introduction, we study properties that can be written as a union of “types,” where each type corresponds to a row distribution  $\tau$  of a matrix  $M$ . The following definitions make this notion precise.

► **Definition 11** (Row-permutation invariant collection of matrices). *A collection of matrices  $\mathcal{B} \subseteq \Sigma^{n \times b}$  is row-permutation invariant if, given a matrix  $B \in \mathcal{B}$ , every row permutation of  $B$  (that is, a matrix resulting from applying the same coordinate permutation to each column of  $B$ ) also belongs to  $\mathcal{B}$ .*

► **Definition 12** (Local and row-symmetric properties). *Let  $\mathcal{P} = \{P_n\}_{n \in \mathbb{N}}$  be a monotone-increasing property, and let  $M_n$  denote the minimal set of  $P_n$ .*

- *If there exists some  $b \in \mathbb{N}$  such that  $M_n \subseteq \Sigma^{n \times b}$  for every  $n$ , we say that  $\mathcal{P}$  is  $b$ -local.*
- *If every  $M_n$  is row-permutation invariant, we say that  $\mathcal{P}$  is row-symmetric.*

► **Remark 13.** Every monotone-increasing property is trivially *column-symmetric*, in the sense that permuting the columns of a matrix in  $M_n$  results in another matrix in  $M_n$ . This naturally reflects the fact that containment of a matrix does not depend on the ordering of the columns, and follows immediately from the definition of a minimal set.

Let  $B \in \Sigma^{n \times b}$ , and consider the collection  $\mathcal{B}$  of all row-permutations of  $B$ . Let  $\tau_B$  denote the *row-distribution* of  $B$ . That is,  $\tau$  is the probability distribution, over  $\Sigma^b$ , of the row  $B_{i,\star}$ , where  $i$  is sampled uniformly from  $[n]$ . Observe that every matrix in  $\mathcal{B}$  has the same row-distribution as  $B$ . Moreover,  $\mathcal{B}$  can be characterized as the set of all matrices with the row distribution  $\tau_B$ . These observations motivate the following definitions.

► **Definition 14** (Type of a matrix). *Let  $B \in \Sigma^{n \times b}$ . We define its type  $\tau_B$  as the distribution of a uniformly random row of  $B$ . That is,  $\tau_B$  is the distribution over  $\Sigma^b$ , such that*

$$\tau_B(x) = \frac{|\{i \in [n] \mid B_i = x\}|}{n}$$

for every  $x \in \Sigma^b$ . Let

$$\mathcal{T}_b^n = \{\tau_B \mid B \in \Sigma_{\text{distinct}}^{n \times b}\}$$

denote the set of all possible types of  $n \times b$  matrices with distinct columns. Given  $\tau \in \mathcal{T}_b^n$ , we denote

$$M_\tau = \{B \in \Sigma^{n \times b} \mid \tau_B = \tau\}.$$

## 5:10 Sharp Threshold Rates for Random Codes

► **Remark 15.** The type of a matrix  $B \in \Sigma^{n \times b}$  determines whether  $B \in \Sigma_{\text{distinct}}^{n \times b}$ . Therefore, for  $\tau \in \mathcal{T}_b^n$ ,

$$M_\tau = \{B \in \Sigma_{\text{distinct}}^{n \times b} \mid \tau_B = \tau\}.$$

The following fact now follows from the above discussion.

► **Fact 16** (Decomposition of a row-permutation invariant collection). *Let  $\mathcal{B} \subseteq \Sigma^{n \times b}$  be a row-permutation invariant collection. Then, there exists a set of types  $T \subseteq \mathcal{T}_{n,b}$  such that*

$$\mathcal{B} = \bigcup_{\tau \in T} M_\tau.$$

Note that a type in  $\mathcal{T}_b^n$  is defined by the number of occurrences of each of  $|\Sigma^b|$  possible rows, in a matrix consisting of  $n$  rows. In particular, each row occurs between 0 and  $n$  times. Thus,

$$|\mathcal{T}_b^n| \leq (n+1)^{|\Sigma^b|} = (n+1)^{q^b}.$$

Crucially for our purposes, this upper bound is polynomial in  $n$ .

### 2.4 Symmetric properties and convex approximations

► **Definition 17.** Let  $\mathcal{T}_b$  denote the simplex of all probability distributions over  $\Sigma^b$ .

It is generally more convenient to work in  $\mathcal{T}_b$  rather than  $\mathcal{T}_b^n$ , since the former is continuous, while the latter is discrete and involves certain divisibility conditions. This motivates the following definition.

► **Definition 18** (Permutation-closed type sets). *A set  $T \subseteq \mathcal{T}_b$  is called permutation-closed if for every  $\tau \in T$  and every permutation  $\pi : [b] \rightarrow [b]$ , the distribution of  $\pi(u)$  (where  $u \sim \tau$ ) also belongs to  $T$ .*

► **Definition 19** (Approximating sets of types). *Fix  $b \in \mathbb{N}$ . Let  $\{T^n\}_{n \in \mathbb{N}}$  be a sequence of sets of types, such that  $T^n \subseteq \mathcal{T}_b^n$ . A (topologically) closed and permutation-closed set  $T \subseteq \mathcal{T}_b$  is an approximation for  $\{T^n\}_{n \in \mathbb{N}}$  if  $T^n \subseteq T$  for every  $n$ , and*

$$\lim_{n \rightarrow \infty} \max_{\tau \in T^n} d_\infty(\tau, T^n) = 0.$$

► **Definition 20** (Symmetric property and convex approximation). *Let  $\mathcal{P} = \{P_n\}_{n \in \mathbb{N}}$  be a  $b$ -local, row-symmetric, monotone-increasing property. Due to Fact 16, for every  $n$  there exists a set  $T_n \subseteq \mathcal{T}_{n,b}$  such that the minimal set of  $P_n$  is  $\bigcup_{\tau \in T_n} M_\tau$ . If the sequence  $\{T_n\}_{n \in \mathbb{N}}$  has a convex approximation  $T$ , we say that  $T$  is a convex approximation for  $\mathcal{P}$ . In this case, we say that  $\mathcal{P}$  is symmetric.*

### 2.5 Non-list-recoverability as a property

Our motivating property is that of being *not* list-recoverable. In this section, we show that non- $(p, \ell, L)$ -list-recoverability is a symmetric property, and we define the convex set  $T_{p,\ell,L}$  that is a convex approximation for it.

Fix  $p \in [0, 1]$ ,  $1 \leq \ell \leq q$  and  $L \in \mathbb{N}$ . Let  $\mathcal{P} = (P_n)_{n \in \mathbb{N}}$  denote the property of being *not*  $(p, \ell, L)$ -list-recoverable. That is, a code  $C \subseteq \Sigma^n$  satisfies  $P_n$  if it contains a  $(p, \ell, L)$ -bad matrix. We now show that  $\mathcal{P}$  is a symmetric property.

Clearly,  $\mathcal{P}$  is monotone-increasing, and its minimal set is exactly the set of  $(p, \ell, L)$ -bad matrices, which we denote  $M_n \subseteq \Sigma_{\text{distinct}}^{n \times L}$ . It follows immediately that  $\mathcal{P}$  is  $L$ -local. Furthermore since the left-hand side of (6) is invariant to row-permutations of  $B$ , the collection  $M_n$  is row-permutation invariant, and so  $\mathcal{P}$  is row-symmetric.

Fact 16 says that we can write  $M_n = \bigcup_{\tau \in T_{p,\ell,L}^n} M_\tau$  for some  $T_{p,\ell,L}^n \subseteq \mathcal{T}_L^n$ . Indeed, (6) yields the following description of  $\mathcal{T}_{p,\ell,L}^n$ : A type  $\tau \in \mathcal{T}_L^n$  belongs to  $\mathcal{T}_{p,\ell,L}^n$  if and only if there exists a distribution  $\rho$  over  $\Sigma^L \times \binom{\Sigma}{\ell}$  such that, given  $(u, K) \sim \rho$ , the following holds:

1. The distribution of  $u$  is  $\tau$ .
2. For every  $1 \leq j \leq L$ , it holds that  $\Pr[u_j \notin K] \leq p$ .
3.  $n \cdot \rho((u, K)) \in \mathbb{N}$  for every  $u$  and  $K$ .

To see this, let  $\rho$  be the joint distribution  $(B_i, K_i)$  for  $i$  uniformly sampled from  $[n]$ , where  $B$  and  $K$  are as in (6). Note that  $\rho$  must satisfy the three conditions above. In the other direction, it is not hard to see that any such distribution  $\rho$  as above gives rise to a matrix of type  $\tau$ , satisfying (6).

We next construct a convex approximation for  $\mathcal{P}$ . Let  $T_{p,\ell,L}$  denote the set of all types  $\tau \in \mathcal{T}_L$  for which there exists a distribution  $\rho$  satisfying Conditions 1 and 2, but not necessarily Condition 3:

► **Definition 21.** Let  $1 \leq \ell \leq L$ ,  $L \in \mathbb{N}$  and  $0 \leq p \leq 1$ . Let  $\tau$  be a distribution over  $\Sigma^L$ . We say that  $\tau$  belongs to the set  $T_{p,\ell,L}$  if there exists a distribution  $\rho$  over  $\Sigma^L \times \binom{\Sigma}{\ell}$  such that:

1. If  $(u, K) \sim \rho$  then the vector  $u$  is  $\tau$ -distributed.
2. For every  $1 \leq j \leq L$  it holds that

$$\Pr_{(u,K) \sim \rho}[u_j \notin K] \leq p.$$

Clearly,  $T_{p,\ell,L}^n \subseteq T_{p,\ell,L}$  for all  $n \in \mathbb{N}$ . It is also immediate to verify that  $T_{p,\ell,L}$  is permutation-closed.

► **Lemma 22.** The set  $T_{p,\ell,L}$  is convex.

**Proof.** Let  $\tau_0, \tau_1 \in T_{p,\ell,L}$ . Let  $t \in [0, 1]$  and let  $\tau_t$  denote the mixture distribution  $(1-t)\tau_0 + t\tau_1$ . Let  $\rho_0$  and  $\rho_1$  be distributions over  $\Sigma^L \times \binom{\Sigma}{\ell}$ , satisfying Conditions 1 and 2 for  $\tau_0$  and  $\tau_1$ , respectively. Let  $\rho_t$  be the mixture distribution  $(1-t)\rho_0 + t\rho_1$ . It is straightforward to verify that  $\rho_t$  satisfies Conditions 1 and 2 with respect to  $\tau_t$ . Hence,  $\tau_t \in T_{p,\ell,L}$ . ◀

The following lemma, proven in the appendix of the full version of this paper, shows that  $T_{p,\ell,L}$  satisfies (19). Namely, every type in  $T_{p,\ell,L}$  can be realized with low error as a type from  $T_{p,\ell,L}^n$ , for large enough  $n$ .

► **Lemma 23.**

$$\lim_{n \rightarrow \infty} \sup_{\tau \in T_{p,\ell,L}} d_\infty(\tau, T_{p,\ell,L}^n) = 0.$$

We record the results of this section in the following corollary.

► **Corollary 24.** Being not  $(p, \ell, L)$ -list-recoverable is a symmetric property. Furthermore,  $T_{p,\ell,L}$  is a convex approximation for this property.

### 3 Characterization theorem

In this section, we prove our main characterization theorem, Theorem 1, which is formally stated below as Theorem 30. Before stating and proving the theorem, we record a few useful lemmas.

► **Lemma 25** ([2, Lemma 2.2]). *Let  $\tau \in \mathcal{T}_b^n$ . Then,*

$$q^{H(\tau)n} \cdot n^{-O_{q,b}(1)} \leq |M_\tau| \leq q^{H(\tau)n}.$$

► **Lemma 26.** *Let  $M \subseteq \Sigma^{n \times b}$ . Then,*

$$|M| \leq n^{q^b} \cdot q^{n \cdot \max_{B \in M} H(\tau_B)}.$$

**Proof.** Let  $T = \{\tau_B \mid B \in M\}$ . Note that

$$M \subseteq \bigcup_{\tau \in T} M_\tau.$$

Thus,

$$|M| \leq \sum_{\tau \in T} |M_\tau| \leq |T| \cdot \max_{\tau \in T} |M_\tau| \leq |\mathcal{T}_{n,b}| \cdot \max_{\tau \in T} |M_\tau|.$$

The claim follows from (2.3) and Lemma 25. ◀

We say that a type is a *histogram type* if it is indifferent to the ordering of a given vector's entries, and thus, only cares about the histogram of the vector. Formally, we make the following definition.

► **Definition 27** (Histogram type). *A type  $\tau \in T_b$  is called a histogram-type if  $\tau(u) = \tau(\pi(u))$  for every  $u \in \Sigma^b$  and every permutation  $\pi : [b] \rightarrow [b]$ .*

► **Lemma 28.** *Let  $T \subseteq T_b$  be a closed, permutation-closed, convex, set of types. Then there exists a histogram type  $\tau \in T$  such that  $H(\tau) = \max_{\tau' \in T} H(\tau')$ .*

**Proof.** Since  $T$  is closed and bounded, it is compact. Thus, there is some  $\tau' \in T$  such that  $H(\tau')$  is maximal. Given a permutation  $\pi : [b] \rightarrow [b]$ , let  $\pi(\tau')$  denote the distribution of the vector  $\pi(u)$ , where  $u \sim \tau'$ . Let

$$\tau = \frac{\sum_{\pi \in \text{Sym}_b} \pi(\tau')}{b!}.$$

Since  $T$  is permutation-closed and convex,  $\tau \in T$ . By concavity of entropy,

$$H(\tau) \geq \frac{\sum_{\pi \in \text{Sym}_b} H(\pi(\tau'))}{b!} = \frac{\sum_{\pi \in \text{Sym}_k} H(\tau')}{b!} = H(\tau').$$

Thus,  $\tau$  has maximum entropy in  $T$ , and is clearly a histogram-type. ◀

The following technical lemma, proven in the appendix of the full version, facilitates our use of an approximation for a set of types.

► **Lemma 29.** *Let  $\tau, \tau' \in T_b$  such that  $d_\infty(\tau, \tau') \leq \varepsilon$ . Then,*

$$|H_{u \sim \tau}(u \mid u_I) - H_{u \sim \tau'}(u \mid u_I)| \leq O_{b,q} \left( \varepsilon \cdot \log \frac{1}{\varepsilon} \right)$$

for any  $I \subseteq [b]$ .

We now prove that every monotone-increasing, local and row-symmetric property with a convex approximation is sharp for random codes. Furthermore, we identify the threshold rate as the maximal entropy in the approximating set.

► **Theorem 30.** Fix  $b \in \mathbb{N}$ . Let  $\mathcal{P} = \{P_n\}_{n \in \mathbb{N}}$  be a symmetric property with locality parameter  $b$ , and let  $T$  be a convex approximation for  $\mathcal{P}$ . Denote  $R^* = 1 - \frac{\max_{\tau \in T} H(\tau)}{b}$ . Fix  $\varepsilon > 0$  and let  $R \in [0, 1]$ . The following now holds.

1. If  $R \leq R^* - \varepsilon$  then

$$\lim_{n \rightarrow \infty} \Pr[C_{\text{RC}}^n(R) \text{ satisfies } \mathcal{P}] = 0.$$

2. If  $R \geq R^* + \varepsilon$  then

$$\lim_{n \rightarrow \infty} \Pr[C_{\text{RC}}^n(R) \text{ satisfies } \mathcal{P}] = 1.$$

**Proof.** For  $b \in \mathbb{N}$  and a matrix  $B \in \Sigma_{\text{distinct}}^{b \times n}$ , let  $X_B$  be an indicator variable for the event that  $B \in C_{\text{RC}}^n(R)$ . For a set  $M \subseteq \Sigma_{\text{distinct}}^{b \times n}$ , let  $X_M = \sum_{B \in M} X_B$ . By Fact 5,

$$\mathbb{E}[X_M] = |M| \cdot q^{-n(1-R)b}.$$

Let  $M_n$  denote the minimal set for  $P_n$  and let  $T_n = \{\tau_B \mid B \in M_n\}$ .

The first statement now follows from Markov's inequality, (3), and Lemma 26:

$$\begin{aligned} \Pr[C \text{ satisfies } \mathcal{P}] &= \Pr[\exists B \in M_n \ B \subseteq C_{\text{RC}}^n(R)] \\ &\leq \Pr[X_M \geq 1] \\ &\leq \mathbb{E}[X_M] \\ &= |M| \cdot q^{-n(1-R)b} \\ &\leq n^{q^b} \cdot q^{n \cdot \max_{\tau \in T_n} H(\tau)} \cdot q^{-n(1-R)b} \\ &\leq n^{q^b} \cdot q^{n \cdot \max_{\tau \in T} H(\tau)} \cdot q^{-n(1-R)b} \\ &\leq n^{q^b} \cdot q^{-nb\varepsilon} \leq e^{-\Omega(n)}. \end{aligned}$$

Above, we used the fact that  $T_n \subseteq T$ .

For the second statement, let  $\tau \in T$  have maximum entropy. By definition 19,  $T$  is closed and permutation-closed, in addition to being convex. Consequently, due to Lemma 28, we may assume that  $\tau$  is a histogram-type. Let  $\tau_n \in T_n$  such that  $d_\infty(\tau, \tau_n) = o_{n \rightarrow \infty}(1)$ . Our plan is to use a second-moment argument to show that  $C_{\text{RC}}^n(R)$  likely contains a matrix of type  $\tau_n$ .

By (3) and Lemma 25,

$$\mathbb{E}[X_{M_{\tau_n}}] = |M_{\tau_n}| q^{-n(1-R)b} \geq q^{(H(\tau_n) - (1-R)b) + o(1)} \geq q^{(H(\tau) - (1-R)b) + o(1)}.$$

We turn to bounding the variance of  $X_{M_{\tau_n}}$ . Fact 5 yields

$$\begin{aligned} \text{Var}[X_{M_{\tau_n}}] &= \sum_{B, B' \in M_{\tau_n}} (\Pr[X_B = X_{B'} = 1] - \Pr[X_B = 1] \Pr[X_{B'} = 1]) \\ &= \sum_{B, B' \in M_{\tau_n}} \left( q^{-n(1-R)(2b - \alpha(B, B'))} - q^{-2n(1-R)b} \right) \\ &\leq \sum_{\substack{B, B' \in M_{\tau_n} \\ \alpha(B, B') \geq 1}} q^{-n(1-R)(2b - \alpha(B, B'))} \end{aligned}$$

where  $\alpha(B, B')$  is the number of columns in  $B'$  that also appear in  $B$ .

## 5:14 Sharp Threshold Rates for Random Codes

In order to bound this sum, we need an estimate on the number of pairs  $B, B'$  with a given  $\alpha(B, B')$ . For  $0 \leq r \leq b$ , let

$$W_r = \{(B, B') \mid B, B' \in M_{\tau_n} \text{ and } \alpha(B, B') = r\}$$

and denote  $S_r = \{\tau_{B\|B'} \mid (B, B') \in W_r\}$ . Here,  $B\|B'$  is the  $n \times 2b$  matrix whose first (resp. last)  $b$  columns are  $B$  (resp.  $B'$ ). By Lemma 26,

$$|W_r| \leq n^{2q^b} \cdot q^{n \max_{\nu \in S_r} H(\nu)}.$$

Let  $(B, B') \in W_r$  and let  $\nu = \tau_{B\|B'}$ . Assume without loss of generality that the first  $r$  columns of  $B$  are identical to the first  $r$  columns of  $B'$ . Let  $u \sim \nu$ . Note that, since  $B, B' \in M_{\tau_n}$ , the random variables  $u_{[b]}$  and  $u_{[2b] \setminus [b]}$  are both  $\tau_n$ -distributed. Hence,

$$\begin{aligned} H(\nu) &= H(u) = H(u_{[2b] \setminus [b]}) + H(u_{[b]} \mid u_{[2b] \setminus [b]}) = H(\tau_n) + H(u_{[b]} \mid u_{[2b] \setminus [b]}) \\ &\leq H(\tau_n) + H(u_{[b]} \mid u_{[r]}) = H(\tau_n) + H(u_{[b] \setminus [r]} \mid u_{[r]}). \end{aligned}$$

Lemma 29 yields

$$\begin{aligned} H(u_{[b] \setminus [r]} \mid u_{[r]}) &\leq H_{v \sim \tau}(v_{[b] \setminus [r]} \mid v_{[r]}) + o(1) \\ &= \sum_{i=r+1}^b H_{v \sim \tau}(v_i \mid v_{[i-1]}) + o(1) \\ &= \sum_{i=r+1}^b H_{v \sim \tau}(v_b \mid v_{[i-1]}) + o(1), \end{aligned}$$

where the last equality is due to  $\tau$  being a histogram-type. Writing

$$f(r) = \sum_{i=r+1}^b H_{v \sim \tau}(v_b \mid v_{[i-1]}),$$

we conclude that

$$H(\nu) \leq f(r) + H(\tau) + o(1),$$

so that

$$|W_r| \leq q^{(f(r) + H(\tau))n + o(n)},$$

and

$$\begin{aligned} \text{Var}[X_{M_{\tau_n}}] &\leq \sum_{r=1}^b |W_r| \cdot q^{-n(1-R)(2b-r)} \leq \sum_{r=1}^b q^{(f(r) + H(\tau) - (1-R)(2b-r))n + o(n)} \\ &\leq \max_{1 \leq r \leq b} q^{(f(r) + H(\tau) - (1-R)(2b-r))n + o(n)}. \end{aligned}$$

By Chebyshev's inequality,

$$\Pr[X_{M_{\tau_n}} = 0] \leq \frac{\text{Var}[X_{M_{\tau_n}}]}{\mathbb{E}[X_{M_{\tau_n}}]^2} \leq \max_{1 \leq r \leq b} q^{(f(r) - H(\tau) + r(1-R))n + o_{b,q}(n)}.$$

We claim that  $(f(r))_{r=0}^b$  is a convex sequence. Indeed,

$$f(r-1) + f(r+1) - 2f(r) = H_{v \sim \tau}(v_b \mid v_{[r-1]}) - H_{v \sim \tau}(v_b \mid v_{[r]}) \geq 0.$$



Therefore, the maximum in the right-hand side of (3) is achieved either by  $r = 1$  or  $r = b$ . In the former case, note that

$$\begin{aligned} f(1) &= \sum_{i=2}^b H_{v \sim \tau}(v_b \mid v_{[i-1]}) = \sum_{i=2}^b H_{v \sim \tau}(v_i \mid v_{[i-1]}) = H_{v \sim \tau}(v \mid v_1) \\ &\leq H(\tau) - H_{v \sim \tau}(v_1) \leq H(\tau) \cdot \frac{b-1}{b}. \end{aligned}$$

In the last inequality above, we used the fact that  $H_{v \sim \tau} v_1 = H_{v \sim \tau} v_i$  for all  $i \in [b]$ , due to  $\tau$  being a histogram-type. Thus, for  $r = 1$ , the corresponding exponent in (3) is

$$(f(1) - H(\tau) + (1 - R))n \leq \left( (1 - R) - \frac{H(\tau)}{b} \right) n \leq -\varepsilon n.$$

In the latter case, since  $f(b) = 0$ , the exponent is

$$(-H(\tau) + (1 - R)b)n \leq -\varepsilon b n.$$

We conclude that

$$\Pr[C_{\text{RC}}^n(R) \text{ does not satisfy } \mathcal{P}] \leq \Pr(X_{M_p} = 0) \leq q^{-\varepsilon n + o(n)}.$$

### Applying the framework to list-recovery

In the rest of the paper, we use Theorem 30 to compute the threshold rate for  $(p, \ell, L)$  list-recovery in several different settings. In order to do that, we set up a few useful definitions.

► **Definition 31** ( $\beta(p, \ell, L)$  and  $\bar{T}_{p, \ell, L}$ ). *Given  $L \in \mathbb{N}$ ,  $\ell \leq L$  and  $p \in [0, 1]$ , let  $\bar{T}_{p, \ell, L}$  denote the set of all histogram-types in  $T_{p, \ell, L}$ . Let*

$$\beta(p, \ell, L) = \max_{\tau \in \bar{T}_{p, \ell, L}} H(\tau).$$

Theorem 30 allows us to characterize the threshold rate for  $(p, \ell, L)$ -list recovery in terms of  $\beta(p, \ell, L)$ :

► **Corollary 32.** *Fix  $L \in \mathbb{N}$ ,  $\ell \leq L$  and  $p \in [0, 1]$ . The threshold rate for  $(p, \ell, L)$  list-recovery is*

$$R^* = 1 - \frac{\beta(p, \ell, L)}{L}.$$

**Proof.** By Corollary 24 and Lemma 28,

$$\beta(p, \ell, L) = \max_{\tau \in T_{p, \ell, L}} H(\tau).$$

The claim now follows from Corollary 24 and Theorem 30. ◀

Finally, we introduce the following notation, which will be used for the rest of the paper.

► **Definition 33** ( $P_\ell(\cdot)$  and  $D_{d, \ell, L}$ ). *Fix  $\ell \leq L$ . Given a vector  $v \in \Sigma^L$  let*

$$P_\ell(v) = \min_{A \in \binom{[L]}{\ell}} |\{i \in [L] \mid v_i \notin A\}|$$

*We use the notation  $D_{d, \ell, L} = \{v \in \Sigma^L \mid P_\ell(v) = d\}$ .*

#### 4 Bounds on the threshold rate for noisy list-recovery

The main result in this section is an estimate of  $\beta(p, \ell, L)$  (Proposition 37 below), which leads to an estimate on the threshold rate for list-recovery (Corollary 38). This estimate is very sharp when  $\frac{q \log L}{L}$  is small; in subsequent sections we will derive estimates which are more precise for certain parameter regimes.

Before coming to these bounds, we begin with a few useful lemmas that bound  $|D_{d,\ell,L}|$  and characterize  $\bar{T}_{p,\ell,L}$ .

► **Lemma 34.** *Let  $r = 1 - \frac{\ell}{q}$  and  $s = \frac{d}{L}$ . Suppose that  $s < r$ . Then,*

$$\binom{q}{rq} \binom{L}{sL} \underbrace{\left( \frac{(1-s)L}{(1-r)q}, \dots, \frac{(1-s)L}{(1-r)q} \right)}_{\ell} \underbrace{\left( \frac{sL}{rq}, \dots, \frac{sL}{rq} \right)}_{q-\ell} \leq$$

$$|D_{d,\ell,L}| \leq \binom{q}{rq} \cdot \left( \sum_{i=0}^{sL} \binom{L}{i} \cdot ((1-r)q)^{L-i} \cdot (rq)^i \right).$$

Using Stirling's approximation, Lemma 34 immediately yields the following.

► **Corollary 35.** *In the setting of Lemma 34, suppose that  $s < r$ . Then,*

$$\log_q |D_{d,\ell,L}| = L(1 - D_{\text{KL}q}(s \parallel r)) \pm O(q \log L),$$

where the underlying constant is universal.

In order to compute  $\beta(p, \ell, L)$ , we will make use of the following characterization of  $\bar{T}_{p,\ell,L}$  (Definition 31). Intuitively, this lemma says that a histogram-type  $\tau$  is bad for  $(p, \ell, L)$ -list-recovery if and only if it has many symbols inside the most frequent  $\ell$  symbols in expectation.

► **Lemma 36.** *Let  $1 \leq \ell \leq q$ ,  $L \in \mathbb{N}$  and  $0 \leq p \leq 1$ . Let  $\tau$  be a distribution over  $\Sigma^L$  and suppose that  $\tau$  is a histogram-type. Then,  $\tau \in \bar{T}_{p,\ell,L}$  if and only if*

$$\mathbb{E}_{u \sim \tau} [P_\ell(u)] \leq pL.$$

Now, we come to our estimate on the threshold rate for  $(p, \ell, L)$  list-recovery in the regime where  $L \rightarrow \infty$  and  $q \leq o(\frac{\log L}{L})$ . We begin with the following proposition, which bounds the quantity  $\beta(p, \ell, L)$ .

► **Proposition 37.** *Let  $r = 1 - \frac{\ell}{q}$  and suppose that  $p \leq r$ . Then,*

$$\beta(p, \ell, L) = L(1 - D_{\text{KL}q}(p \parallel r)) \pm O(q \log L).$$

► **Corollary 38.** *The threshold rate for  $(p, \ell, L)$  list-recovery of a random code is*

$$R^* = \begin{cases} D_{\text{KL}q}(p \parallel r) \pm O\left(\frac{q \log L}{L}\right) & \text{if } p < r \\ 0 & \text{if } p \geq r, \end{cases}$$

where  $r = 1 - \frac{\ell}{q}$ .

► **Remark 39.** To make sense of the threshold in Corollary 38, one can verify the identity

$$D_{\text{KL}_q}(p \parallel 1 - \ell/q) = 1 - p \log_q \left( \frac{q - \ell}{p} \right) - (1 - p) \log_q \left( \frac{\ell}{1 - p} \right).$$

Substituting  $\ell = 1$ , we find  $D_{\text{KL}_q}(p \parallel 1 - 1/q) = 1 - h_q(p)$ , agreeing with the list decoding capacity theorem. For larger  $\ell$ , this expression agrees with the *list-recovery capacity theorem*, as stated in e.g. [14].

## 5 Zero-error list-recovery and perfect hashing codes

In this section we analyze the threshold rate for zero-error list-recovery (that is, when  $p = 0$ ), and give a more precise version of Corollary 38 in this setting.

► **Lemma 40.** *Let  $p^* = |D_{0,\ell,L}|/q^L$ . The threshold rate for  $(0, \ell, L)$  list-recovery of a random code is*

$$R^* = \frac{-\log_q(p^*)}{L}.$$

We use this to compute the threshold rate for a random code to be a perfect hash code, which is the same as being  $(0, q - 1, q)$  list-recoverable.

► **Corollary 41.** *The threshold rate for  $(0, q - 1, q)$  list-recovery of a random code is*

$$R^* = \frac{1}{q} \log_q \left( \frac{1}{1 - q!/q^q} \right).$$

The corollary follows from the lemma in a straightforward manner by verifying that  $|D_{0,q-1,q}| = q^q - q!$ .

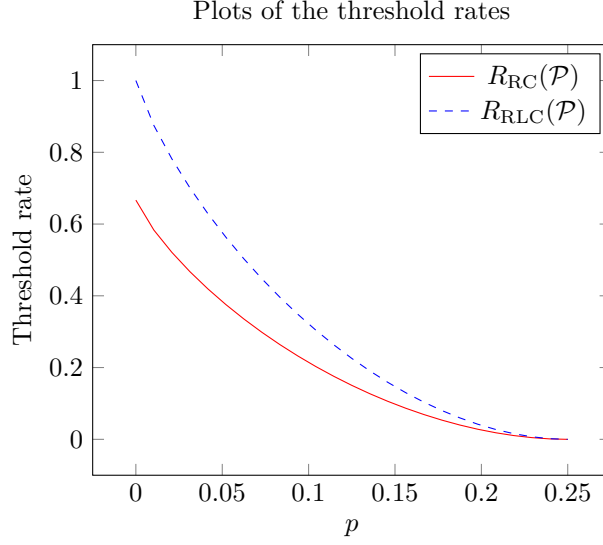
## 6 List of two decoding of random and random linear codes

In this section, we study the list-of-2 decodability of two random ensembles of codes. In detail, we precisely compute the threshold rate for  $(p, 3)$ -list-decoding for random codes and for random *linear* codes. Denote by  $\mathcal{P}$  the monotone increasing property of *not* being  $(p, 3)$ -list-decodable. Note that we cannot immediately apply Corollary 38, as the error term of  $O\left(\frac{q \log L}{L}\right)$  is not negligible in this regime. We specialize to the case of  $q = 2$ , and recall our convention that  $\log$  denotes the base-2 logarithm. Recall from the introduction that whenever  $p < 1/4$  there exist  $(p, 3)$ -list-decodable codes with positive rate, but whenever  $p > 1/4$  the only  $(p, 3)$ -list-decodable codes are of bounded size, independent of  $n$ .

Our main result of this section is a demonstration that the list-of-2 decoding threshold rate for random *linear* codes is in fact greater than the corresponding threshold rate for random codes. This result demonstrates that our techniques are precise enough to allow us to sharply delineate between different natural ensembles of codes.

In the following,  $C_{\text{RLC}}^n(R)$  denotes a random linear code of block length  $n$  and rate  $R$ . We define the threshold rate for random linear codes in a manner analogous to the definition for random codes:

$$R_{\text{RLC}}^n(\mathcal{P}) := \begin{cases} \sup \{R \in [0, 1] : \Pr[C_{\text{RLC}}^n(R) \text{ satisfies } \mathcal{P}] \leq \frac{1}{2}\} & \text{if there is such an } R \\ 0 & \text{otherwise.} \end{cases}$$



■ **Figure 1** The threshold rate  $R_{RC}$  (red) for  $(p, 3)$ -list-decodability of random codes, and the threshold rate  $R_{RLC}$  (blue, dashed) for  $(p, 3)$ -list-decodability of random *linear* codes. Note that, uniformly over  $p$ , random linear codes have the greater threshold rate.

► **Theorem 42.** *Let  $p \in (0, 1/4)$ .*

1. *The threshold rate for  $(p, 3)$ -list-decoding for random codes satisfies*

$$\lim_{n \rightarrow \infty} R_{RC}^n(\mathcal{P}) = 1 - \frac{1 + h(3p) + 3p \log 3}{3}.$$

2. *The threshold rate for  $(p, 3)$ -list-decoding for random linear codes satisfies*

$$\lim_{n \rightarrow \infty} R_{RLC}^n(\mathcal{P}) = 1 - \frac{h(3p) + 3p \log 3}{2}.$$

Note that the threshold rate for random linear codes is greater than the threshold rate for random codes, uniformly over  $p \in (0, 1/4)$ . See Figure 1.

## 7 Computing the threshold rate for list-recovery efficiently

In the previous sections, we gave precise analytical expressions for the threshold rate for list-recovery in certain parameter regimes. However, there are some regimes where these bounds aren't precise. In this section, we consider the question of computing the threshold rate  $R^*$  algorithmically, given  $p, \ell$  and  $L$ . We use tools from the study of entropy-maximizing distributions to develop a simple binary-search-based procedure to pinpoint  $R^*$  up to arbitrarily small additive error.

We begin with a lemma that shows that we can compute the cardinality  $|D_{d,\ell,L}|$  efficiently; we will use this as a subroutine in our final algorithm.

► **Lemma 43.** *Given  $0 \leq d \leq L$  and  $1 \leq \ell \leq q$ , the cardinality  $|D_{d,\ell,L}|$  can be computed in time*

$$O((L+1)^q + \text{poly}(q, L)).$$

We recall the following standard facts from the theory of entropy-maximizing distributions.

► **Lemma 44** ([17, Sec. 3]). Let  $\Omega$  be a finite nonempty set,  $f : \Omega \rightarrow \mathbb{R}$  and  $t \in \mathbb{R}$ . Let  $S_t$  denote the set of all distributions  $\tau$  over  $\Omega$  such that  $\mathbb{E}_{\omega \sim \tau} [f(\omega)] = t$ . Let

$$F(t) = \max_{\tau \in S_t} H(\tau).$$

Then

$$F(t) = \inf_{\alpha \in \mathbb{R}} \left[ \log_q \left( \sum_{\omega \in \Omega} q^{\alpha \cdot f(\omega)} \right) - \alpha t \right].$$

Furthermore:

1. If  $\tau$  is the entropy maximizing distribution, then  $\tau(\omega) = \tau(\omega')$  for every  $\omega, \omega' \in \Omega$  such that  $f(\omega) = f(\omega')$ .
2. Let  $t^* = \mathbb{E}_{\omega \sim \text{Uniform}(\Omega)} [f(\omega)]$ . Then,  $F(t^*) = \log |\Omega|$ , and  $F(t)$  is nondecreasing (resp. nonincreasing) in the range  $t < t^*$  (resp.  $t > t^*$ ).
3. The function

$$\log_q \left( \sum_{\omega \in \Omega} q^{\alpha \cdot f(\omega)} \right) - \alpha t$$

is convex in  $\alpha$ .

► **Lemma 45.** Let  $\ell \leq q$ ,  $L \in \mathbb{N}$  and  $0 < p \leq 1$ , and let  $t^* = q^{-L} \cdot \sum_{d=0}^L d \cdot |D_{d,\ell,L}|$ . Then,

$$\beta(p, \ell, L) = \begin{cases} \inf_{\alpha \in \mathbb{R}} \left[ \log_q \left( \sum_{d=0}^L |D_{d,\ell,L}| \cdot q^{\alpha d} \right) - \alpha p L \right] & \text{if } pL < t^* \\ L & \text{if } pL \geq t^*. \end{cases}$$

► **Remark 46.** In general,  $\frac{t^*}{L}$  is slightly smaller than  $1 - \frac{\ell}{q}$ . Thus, Lemma 45 extends the range in which the threshold is 0 from  $\left[1 - \frac{\ell}{q}, 1\right]$  (Corollary 38) to  $[t^*, 1]$ .

► **Corollary 47.** There is an algorithm, that, given  $p, \ell, L$  and  $\varepsilon > 0$ , computes the threshold-rate for  $(p, \ell, L)$ -list-recovery, within an additive error of  $\varepsilon$ . The runtime of this algorithm is  $O((L+1)^q + \text{poly}(q, L, \log \frac{1}{\varepsilon}, \beta(p)))$ , where

$$\beta(p) = \begin{cases} \log \frac{1}{p} & \text{if } p > 0 \\ 1 & \text{if } p = 0. \end{cases}$$

---

## References

- 1 Béla Bollobás. *Random Graphs, Second Edition*, volume 73 of *Cambridge Studies in Advanced Mathematics*. Cambridge University Press, 2001. doi:10.1017/CB09780511814068.
- 2 Imre Csiszár and Paul C Shields. *Information theory and statistics: A tutorial*. Now Publishers Inc, 2004.
- 3 P. Erdős and A. Rényi. On random graphs. I. *Publ. Math. Debrecen*, 6:290–297, 1959.
- 4 Keith Frankston, Jeff Kahn, Bhargav Narayanan, and Jinyoung Park. Thresholds versus fractional expectation-thresholds. *arXiv preprint*, 2019. arXiv:1910.13433.
- 5 Michael L Fredman and János Komlós. On the size of separating systems and families of perfect hash functions. *SIAM Journal on Algebraic Discrete Methods*, 5(1):61–68, 1984.
- 6 Ehud Friedgut. Sharp thresholds of graph properties, and the  $k$ -sat problem. *J. Amer. Math. Soc.*, 12(4):1017–1054, 1999. With an appendix by Jean Bourgain. doi:10.1090/S0894-0347-99-00305-7.

- 7 Venkatesan Guruswami, Johan Håstad, and Swastik Kopparty. On the list-decodability of random linear codes. *IEEE Trans. Information Theory*, 57(2):718–725, 2011. doi:10.1109/TIT.2010.2095170.
- 8 Venkatesan Guruswami, Ray Li, Jonathan Mosheiff, Nicolas Resch, Shashwat Silas, and Mary Wootters. Bounds for list-decoding and list-recovery of random linear codes. *arXiv preprint*, 2020. arXiv:2004.13247.
- 9 Venkatesan Guruswami and Andrii Riazanov. Beating fredman-komlós for perfect k-hashing. In *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2019.
- 10 Venkatesan Guruswami, Christopher Umans, and Salil Vadhan. Unbalanced expanders and randomness extractors from parvaresh–vardy codes. *Journal of the ACM (JACM)*, 56(4):1–34, 2009.
- 11 J Korner and Katalin Marton. New bounds for perfect hashing via information theory. *European Journal of Combinatorics*, 9(6):523–530, 1988.
- 12 Jénos Körner. Fredman–komlós bounds and information theory. *SIAM Journal on Algebraic Discrete Methods*, 7(4):560–570, 1986.
- 13 Jonathan Mosheiff, Nicolas Resch, Noga Ron-Zewi, Shashwat Silas, and Mary Wootters. Ldpc codes achieve list decoding capacity. *arXiv preprint*, 2019. arXiv:1909.06430.
- 14 Nicolas Resch. *List-Decodable Codes:(Randomized) Constructions and Applications*. PhD thesis, Carnegie Mellon University, 2020.
- 15 Atri Rudra and Mary Wootters. Average-radius list-recovery of random linear codes. In *Proceedings of the 2018 ACM-SIAM Symposium on Discrete Algorithms, SODA*, 2018.
- 16 Salil P. Vadhan. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 7(1-3):1–336, 2012. doi:10.1561/04000000010.
- 17 Martin J. Wainwright and Michael I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305, 2008. doi:10.1561/22000000001.
- 18 Chaoping Xing and Chen Yuan. Beating the probabilistic lower bound on perfect hashing. *arXiv preprint*, 2019. arXiv:1908.08792.
- 19 Victor Vasilievich Zyablov and Mark Semenovich Pinsker. List concatenated decoding. *Problemy Peredachi Informatsii*, 17(4):29–33, 1981.

# Simple Heuristics Yield Provable Algorithms for Masked Low-Rank Approximation

**Cameron Musco**

College of Information and Computer Sciences, University of Massachusetts Amherst, MA, USA  
cmusco@cs.umass.edu

**Christopher Musco**

Department of Computer Science and Engineering,  
New York University Tandon School of Engineering, NY, USA  
cmusco@nyu.edu

**David P. Woodruff**

Department of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA  
dwoodruf@cs.cmu.edu

---

## Abstract

In the *masked low-rank approximation problem*, one is given data matrix  $A \in \mathbb{R}^{n \times n}$  and binary mask matrix  $W \in \{0, 1\}^{n \times n}$ . The goal is to find a rank- $k$  matrix  $L$  for which:

$$\text{cost}(L) \stackrel{\text{def}}{=} \sum_{i=1}^n \sum_{j=1}^n W_{i,j} \cdot (A_{i,j} - L_{i,j})^2 \leq \text{OPT} + \epsilon \|A\|_F^2,$$

where  $\text{OPT} = \min_{\text{rank-}k \text{ } \hat{L}} \text{cost}(\hat{L})$  and  $\epsilon$  is a given error parameter. Depending on the choice of  $W$ , the above problem captures factor analysis, low-rank plus diagonal decomposition, robust PCA, low-rank matrix completion, low-rank plus block matrix approximation, low-rank recovery from monotone missing data, and a number of other important problems. Many of these problems are NP-hard, and while algorithms with provable guarantees are known in some cases, they either 1) run in time  $n^{\Omega(k^2/\epsilon)}$  or 2) make strong assumptions, for example, that  $A$  is incoherent or that the entries in  $W$  are chosen independently and uniformly at random.

In this work, we show that a common polynomial time heuristic, which simply sets  $A$  to 0 where  $W$  is 0, and then finds a standard low-rank approximation, yields bicriteria approximation guarantees for this problem. In particular, for rank  $k' > k$  depending on the *public coin partition number* of  $W$ , the heuristic outputs rank- $k'$   $L$  with  $\text{cost}(L) \leq \text{OPT} + \epsilon \|A\|_F^2$ . This partition number is in turn bounded by the randomized communication complexity of  $W$ , when interpreted as a two-player communication matrix. For many important cases, including all those listed above, this yields bicriteria approximation guarantees with rank  $k' = k \cdot \text{poly}(\log n/\epsilon)$ .

Beyond this result, we show that different notions of communication complexity yield bicriteria algorithms for natural variants of masked low-rank approximation. For example, multi-player number-in-hand communication complexity connects to masked tensor decomposition and non-deterministic communication complexity to masked Boolean low-rank factorization.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Approximation algorithms analysis; Theory of computation  $\rightarrow$  Communication complexity

**Keywords and phrases** low-rank approximation, communication complexity, weighted low-rank approximation, bicriteria approximation algorithms

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.6

**Related Version** Full paper available at <https://arxiv.org/abs/1904.09841>.

**Funding** *David P. Woodruff*: David Woodruff would like to thank support from the Office of Naval Research (ONR) grant N00014-18-1-2562.

**Acknowledgements** Part of this work was done while the authors were visiting the Simons Institute for the Theory of Computing.



© Cameron Musco, Christopher Musco, and David P. Woodruff;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 6; pp. 6:1–6:20



Leibniz International Proceedings in Informatics

LIPIcs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

## 1 Introduction

The goal of low-rank approximation is to approximate an  $n \times n$  matrix  $A$  with a rank- $k$  matrix  $L$ .  $L$  can be written as the product  $L = U \cdot V$  of a “tall-and-thin” matrix  $U$  and a “short-and-wide” matrix  $V$  with  $k$  columns and rows respectively. For  $k \ll n$  this approximation can lead to computational speedups: one can store the factors  $U$  and  $V$  with less memory than storing  $A$  itself, and can compute the product  $U \cdot V \cdot x$  with a vector  $x$  faster than computing  $A \cdot x$ . Additionally, low-rank approximation is useful for denoising and can reveal low-dimensional structure in high-dimensional data (it is e.g., the basis behind principal component analysis). It thus serves as a preprocessing step in many applications, including clustering, data mining, and recommendation systems. The optimal low-rank approximation to  $A$  with distance measured in the Frobenius, spectral, or any unitarily invariant norm can be computed in polynomial time using a singular value decomposition (SVD). There are also extremely efficient approximation algorithms for finding a near optimal  $L$  under different measures, including the Frobenius norm, spectral norm, and various entrywise norms. For a comprehensive treatment, we refer the reader to the surveys [36, 47, 70].

Despite its wide applicability, in many situations standard low-rank approximation does not suffice. For example, it is common that certain entries in  $A$  either *don't obey underlying low-rank structure* or are *missing*. For example,  $A$  may be close to low-rank but with a small number of corrupted entries, or may be the sum of a low-rank matrix plus a high-rank, but still efficiently representable, diagonal or block diagonal matrix. In both cases, one must compute a low-rank approximation of  $A$  ignoring the outlying entries. One can formalize this problem, considering a binary matrix  $W$  with  $W_{i,j} = 0$  for each outlying entry  $(i, j)$  of  $A$  and  $W_{i,j} = 1$  otherwise.

► **Problem 1 (Masked Low-Rank Approximation).** *Given  $A \in \mathbb{R}^{n \times n}$ , binary  $W \in \{0, 1\}^{n \times n}$ , and rank parameter  $k$ , find rank- $k$   $L$  minimizing:*

$$\|W \circ (A - L)\|_F^2 = \sum_{i,j \in [n]} W_{i,j} \cdot (A_{i,j} - L_{i,j})^2,$$

where for two matrices  $M$  and  $N$  of the same size,  $M \circ N$  denotes the entrywise (Hadamard product): with  $(M \circ N)_{i,j} = M_{i,j} \cdot N_{i,j}$  and for integer  $n$ ,  $[n]$  denotes  $\{1, \dots, n\}$ .

As stated, Problem 1 minimizes the squared Frobenius norm of  $W \circ (A - L)$ . However any matrix norm can be used. In any case, is unclear how to extend standard low-rank approximation algorithms to solving Problem 1, since they optimize over the full matrix  $A$ , without the ability to take into account  $W$  encoding entries that should be ignored. We note that Problem 1 is equivalent to minimizing  $\|A - (L + S)\|_F^2$  where  $L$  is rank- $k$  and  $S$  is any matrix with support restricted to the 0 entries of  $W$ . If these zeros are on the diagonal, then  $S$  is diagonal. If they are sparse, then  $S$  is sparse, etc. This is how Problem 1 is traditionally stated in many applications.

### 1.1 Existing Work

A common approach to solving Problem 1 is to apply alternating minimization or the EM (Expectation-Maximization) algorithm. In fact, factor analysis, a slight variant of Problem 1 when  $W$  is 0 on its diagonal and 1 off the diagonal, was one of the original motivations of the EM algorithm [21, 56]. Much recent work studies when alternating minimization for Problem 1 converges in polynomial time under the assumptions that (1) there is a solution  $L = U \cdot V \approx A$  which is *incoherent*, meaning that the squared row norms of  $U$  and column



norms of  $V$  are small and (2) the entries of  $W$  are selected at random or have pseudorandom properties [73, 40, 51]. Under similar assumptions it can be shown that Problem 1 and the related problem of *robust PCA* can be solved via convex relaxation in polynomial time [13, 71, 12]. In many cases, these algorithms perform well in practice even when the above assumptions do not hold. Additionally, they can be proven to run in polynomial time in some common settings when the entries of  $W$  are not random – e.g., when  $W$  is zero only on its diagonal or at a few arbitrary locations. That is, when we want to approximate  $A$  as a low-rank plus diagonal component, or a low-rank matrix with arbitrary sparse corruptions respectively. However, these results still require assuming the existence of  $U \cdot V$  that is incoherent and further that is *exact* – with  $U \cdot V$

A natural question is if for common mask patterns, one can obtain provable algorithms without incoherence or other strong assumptions. This approach was taken in [54] in the context of *weighted low-rank approximation*, where  $W$  is a nonnegative matrix and the objective is still to minimize  $\|W \circ (A - L)\|_F^2$ . When  $W$  is binary, this reduces to Problem 1. In [54] it was shown that if  $W$  has at most  $r$  distinct columns, then it is possible to obtain a relative error guarantee in  $2^{\text{poly}(rk/\epsilon)} \cdot \text{poly}(n)$  time. More generally, if the rank of  $W$  over the reals is at most  $r$ , then  $n^{\text{poly}(rk/\epsilon)}$  time is achievable. Note that such algorithms are only polynomial time if  $k$ ,  $r$ , and  $1/\epsilon$  are very small. In many common use cases, such as when  $W$  is all 0s on the diagonal and 1 off-diagonal (corresponding to low-rank plus diagonal decomposition), or when  $W$  is all 0s above the diagonal and 1s on or beneath the diagonal,  $r$  is large: in fact  $\text{rank}(W) = r = n$  in these cases.

When  $A$  is low-rank with sparse corruptions, i.e., when  $W$  has at most  $t$  zero entries per row and column, the algorithms of [54] can be applied if there is an exact solution (with  $A = L$  on all non-corrupted entries). [54] referred to this problem as *adversarial matrix completion* and gave an  $n^{O(tk^2)}$  time algorithm. This is only polynomial time for constant values of  $t$  and  $k$ , and even for constant  $t$  and  $k$  is very large. Moreover, their method cannot be used in the approximate case since it requires creating a low-rank weight matrix  $W'$  whose support matches that of  $W$ . Since  $W$  may be far from low-rank, the non-zero entries of  $W$  and  $W'$  necessarily have very different values. This introduces significant error, unless  $A = L$  exactly on the support of  $W$ .

## 1.2 Our Contributions

With the goal of obtaining fast masked low-rank approximation algorithms, we consider *bicriteria approximation* with additive error. That is, we allow the rank  $k'$  of the output  $L$  to be slightly larger than  $k$ , but one still compares to the best rank- $k$  approximation. Formally, given  $A \in \mathbb{R}^{n \times n}$ ,  $W \in \{0, 1\}^{n \times n}$ , and an error parameter  $\epsilon$ , we would like to find a rank- $k'$  matrix  $L$  for which:

$$\|W \circ (A - L)\|_F^2 \leq OPT + \epsilon \|A\|_F^2, \quad (1)$$

where  $OPT = \min_{\text{rank-}k} \hat{L} \|W \circ (A - \hat{L})\|_F^2$  is the optimal value of Problem 1.

Assuming a variant of the Exponential Time Hypothesis, [54] shows a lower bound of  $2^{\Omega(r)}$  time for finding rank- $k$   $L$  achieving (1) with constant  $\epsilon$  when  $W$  is rank- $r$ . Thus the relaxation to bicriteria approximation seems necessary. In many applications it is not essential for the output rank  $k'$  to be exactly  $k$  – as long as  $k'$  is small, one still obtains significant compression. Indeed, bicriteria algorithms for low-rank matrix approximation are widely studied [22, 23, 18, 17, 63, 9]. The starting point of our work is the question:

*For which mask patterns  $W \in \{0, 1\}^{n \times n}$  can one obtain efficient bicriteria low-rank approximation algorithms with  $k' \leq k \cdot \text{poly}((\log n)/\epsilon)$  satisfying (1)?*

## Main Results

We show that the answer to this question is related to the *randomized communication complexity* of  $W$ .<sup>1</sup> If the rows and columns of  $W \in \{0, 1\}^{n \times n}$  are indexed by strings  $x \in \{0, 1\}^{\log n}$  and  $y \in \{0, 1\}^{\log n}$ , respectively, we can think of  $W$  as a two-player communication matrix for a Boolean function  $f$ , where  $f(x, y) = W_{x,y}$ . Here Alice has  $x$ , Bob has  $y$ , and the two parties want to exchange messages with as few bits as possible to compute  $f(x, y)$  with probability at least  $1 - \delta$ . The number of bits required is the randomized communication complexity  $R_\delta(f)$ . If we further require that the protocol never errs when  $f(x, y) = 1$ , but for any fixed pair  $(x, y)$  with  $f(x, y) = 0$ , it errs with probability at most  $\delta$ , then the number of bits required is the 1-sided randomized communication complexity  $R_\delta^{1\text{-sided}}(f)$ . We show:

► **Theorem 1.** *Letting  $f$  be the function computed by  $W \in \{0, 1\}^{n \times n}$  and  $\neg f$  be its negation, there is a bicriteria low-rank approximation  $L$  with rank  $k' = k \cdot 2^{R_\epsilon^{1\text{-sided}}(\neg f)}$  achieving:*

$$\|W \circ (A - L)\|_F^2 \leq OPT + 2\epsilon \|A \circ W\|_F^2,$$

where  $OPT = \min_{\text{rank-}k \hat{L}} \|W \circ (A - \hat{L})\|_F^2$ .  $L$  is computable in  $O(\text{nnz}(A)) + n \cdot \text{poly}(k'/\epsilon)$  time.

As we will see, for many common  $W$ ,  $R_\epsilon^{1\text{-sided}}(\neg f)$  is very small – with  $2^{R_\epsilon^{1\text{-sided}}(\neg f)}$  at most  $\text{poly}(\log n/\epsilon)$ . Note that our additive error is in terms of  $\|A \circ W\|_F^2$  which is only smaller than  $\|A\|_F^2$ , and may be much smaller, if e.g., the zeros in  $W$  correspond to corruptions in  $A$ . We also show a bound in terms of the communication complexity with 2-sided error.

► **Theorem 2.** *Letting  $f$  be the function computed by  $W \in \{0, 1\}^{n \times n}$ , there is a bicriteria low-rank approximation  $L$  with rank  $k' = k \cdot 2^{R_\epsilon(f)}$  achieving:*

$$\|W \circ (A - L)\|_F^2 \leq OPT + 2\epsilon \|A \circ W\|_F^2 + \epsilon \|L_{\text{opt}} \circ (1 - W)\|_F^2,$$

where  $OPT = \min_{\text{rank-}k \hat{L}} \|W \circ (A - \hat{L})\|_F^2$  and  $L_{\text{opt}}$  is any rank- $k$  matrix achieving  $OPT$ .  $L$  is computable in  $O(\text{nnz}(A)) + n \cdot \text{poly}(k'/\epsilon)$  time.

Further, the algorithm achieving Theorems 1 and 2 is extremely simple: just zero out the entries in  $A$  corresponding to entries in  $W$  that are 0 (i.e., compute  $A \circ W$ ), and then output a standard rank- $k'$  approximation of the resulting matrix. This is already a widely-used heuristic for solving Problem 1 [3, 74], and we obtain the first provable guarantees. An optimal low-rank approximation of  $A \circ W$  can be computed in polynomial time via an SVD. An approximation achieving relative error  $(1 + \epsilon)$  can be computed with high probability in  $O(\text{nnz}(A)) + n \cdot \text{poly}(k/\epsilon)$  time, giving the runtime bounds of Theorems 1 and 2 [19].

### 1.2.1 Applications

Theorems 1 and 2 provide the first bicriteria approximation algorithms for Problem 1 with small  $k'$  for a number of important special cases of the mask matrix  $W$ :

1. If  $W$  has at most  $t$  zero entries in each row, this is Low-Rank Plus Sparse (LRPS) matrix approximation, which captures the challenge of finding a low-rank approximation when a few entries are not known, or do not obey underlying low-rank structure. It has been studied in the context of adversarial matrix completion [58], robust matrix decomposition [31, 12], optics, system identification [7], and more [15].

<sup>1</sup> Our bounds actually hold for the public coin partition number of  $W$ , which is upper bounded by the randomized communication complexity [34]. See Section 1.2.4 for a more detailed discussion.

2. If  $W$  is zero exactly on the diagonal entries, this is Low-Rank Plus Diagonal (LRPD) matrix approximation. This problem arises since in practice, many matrices that are not close to low-rank are close to diagonal, or contain a mixture of diagonal and low-rank components [15]. This observation has been used e.g., to construct compact representations of kernel matrices [61, 69], weight matrices in neural networks [48, 74], and covariance matrices [66, 65]. LRPD approximation also arises in applications related to source separation [44] and variational inference [49] and is closely related to factor analysis [64, 57], which adds the additional constraints that  $L$  and  $A - L$  are PSD.
3. If  $W$  is the negation of a block-diagonal matrix with blocks of varying sizes, meaning that  $W$  is 0 on entries in the blocks and 1 on entries outside of the blocks, this is Low-Rank Plus Block-Diagonal (LRPBD) matrix approximation. This is a natural generalization of the LRPD problem and has been studied in the context of anomaly detection in networks [4], foreground detection [29], and robust principal component analysis [41]. We also consider the natural generalization of LRPS approximation discussed above, which we call the Low-Rank Plus Block-Sparse (LRPBS) matrix approximation problem.
4. If each row of  $W$  has a prefix of ones, followed by a suffix of zeros, this is the Monotone Missing Data Pattern (MMDP) problem. This is a common missing data pattern, arising in the event that when a variable is missing from a sample, all subsequent variables are also missing. Methods for handling this pattern are, e.g., included in the SAS/STAT package for statistical analyses [1]. We refer the reader to [68] for more examples of common missing data patterns, such as “connected” and “file matching” patterns.
5. If  $W$  is the negation of a banded matrix where  $W_{i,j} = 0$  iff  $|i - j| < p$  for some distance  $p$ , this is Low-Rank Plus Banded (LRPBand) matrix approximation. Variants of this problem arise in scientific computing and machine learning, in particular in the approximation of kernel matrices via fast multipole methods [55, 28, 72]. These methods approximate a kernel matrix using a low-rank “far-field” component, and a “near-field” component, which explicitly represents the kernel function between close points. If points are in one dimension and sorted, this corresponds to approximating  $A$  with a low-rank plus banded matrix. Many methods compute the low-rank component analytically (using polynomial approximations of the kernel function). A natural alternative is to seek an optimal decomposition via Problem 1. Many applications involve higher dimensional data. E.g., in the two-dimensional case, each  $i \in [n]$  can be mapped to  $(i_1, i_2) \in [\sqrt{n}] \times [\sqrt{n}]$  where  $i_1, i_2$  correspond to the first and second halves of  $i$ 's binary expansion.  $W_{i,j} = 0$  iff  $|i_1 - j_1| + |i_2 - j_2| < p$ . We give similar bounds for this multidimensional variant.

We summarize our results for the above weight patterns in Table 1. We detail the specific functions  $f$  used in these applications in Sections 2 and 3, but note that (1), (2), and (3) use variants of Equality, which has  $O(\log(1/\epsilon))$  randomized 1-sided error communication complexity, (4) and (5) use a variant of the Greater-Than problem with  $O(\log \log n + \log(1/\epsilon))$  randomized 2-sided error communication complexity for  $\log n$  bit inputs.

### 1.2.2 Relation to Matrix Completion

Masked low-rank approximation is closely related to the well-studied matrix completion problem [13, 32, 37], however the goal is different. In masked low-rank approximation, we want to approximate  $A$  as accurately as possible on the *non-masked entries* (i.e., where  $W_{ij} = 1$ ). In matrix completion, the support of  $W$  represents entries in  $A$  that are observed and the goal is to approximate  $A$  on the *missing entries* (i.e., where  $W_{ij} = 0$ ). The most common approach to solving this problem is in fact to find a low-rank approximation fitting the non-missing entries (i.e., to solve Problem 1), however the two problems are not equivalent.

■ **Table 1** Summary of applications of Theorems 1 and 2.

Mask Pattern	$k'$	Communication Problem	Ref.
LRPD/LRPBD	$O(k/\epsilon)$	Equality	Cor. 15 & 16
LRPBand	$k \cdot \text{poly}\left(\frac{\log n}{\epsilon}\right)$	Variant of Greater-Than	Cors. 17
LRPS/LRPBS (w/ sparsity $t$ )	$O(kt/\epsilon)$	Variant of equality	Full paper
MMDP	$k \cdot \text{poly}\left(\frac{\log n}{\epsilon}\right)$	Greater-Than	Cor. 18
Subsampled Toeplitz	$O(\min(pk, k/\epsilon))$	Equality mod $p$	Full paper

For example, it is not clear that a bicriteria solution to Problem 1, as given by Theorems 1 and 2, will give anything interesting for the matrix completion problem. In fact, our proof technique implies that it likely will not.

We further note that in matrix completion, the mask  $W$  is typically assumed to be random and the goal is to recover the missing entries of  $A$  when  $W$  has as few sampled ones as possible. We do not expect that a random matrix will have low-communication complexity, unless it has further structure (e.g., few zeros or ones per row).

### 1.2.3 Other Communication Models

Theorems 1 and 2 connect communication complexity to the analysis of a simple heuristic for masked low-rank approximation. A natural question is:

*Can other notions of communication complexity, such as multi-party communication complexity, non-deterministic communication complexity, and communication complexity of non-Boolean functions yield algorithms for masked low-rank approximation?*

We answer this question affirmatively. We first look at multi-party communication complexity, which we show corresponds to masked tensor low-rank approximation. Here we focus on order-3 tensors, though our results are proven for arbitrary order- $t$  tensors. A tensor is just an array  $A \in \mathbb{R}^{n \times n \times n}$ . In masked low-rank tensor approximation we are given such an  $A$  and a mask tensor  $W \in \{0, 1\}^{n \times n \times n}$  and the goal is to find rank- $k$  tensor  $L$  minimizing  $\|W \circ (A - L)\|_F^2$ . This problem has been widely studied in the context of low-rank tensor completion [25, 43, 50] and robust tensor PCA [42, 45], which corresponds to the setting where  $W$ 's zeros represent sparse corruptions of an otherwise low-rank tensor. Applications include color image and video reconstruction along with low-rank plus diagonal tensor approximation [8], where  $W$  is zero on its diagonal and one everywhere else. In the full paper we show:

► **Theorem 3** (Multipart Communication Complexity  $\rightarrow$  Tensor Low-Rank Approx). *Let  $f$  be the function computed by  $W \in \{0, 1\}^{n \times n \times n}$ ,  $\neg f$  be its negation, and  $R_{\epsilon}^{3,1\text{-sided}}(\neg f)$  be the randomized 3-party communication complexity of  $\neg f$  in the number-in-hand black-board model with 1-sided error. A bicriteria low-rank approximation  $L$  with rank  $k' = O\left((k/\epsilon)^2 \cdot 4^{R_{\epsilon}^{3,1\text{-sided}}(\neg f)}\right)$  achieving:*

$$\|W \circ (A - L)\|_F^2 \leq OPT + 2\epsilon \|A \circ W\|_F^2,$$

where  $OPT = \inf_{\text{rank-}k \hat{L}} \|W \circ (A - \hat{L})\|_F^2$ , can be computed in  $O(\text{nnz}(A)) + n \cdot \text{poly}(k/\epsilon)$  time.

We give applications of Theorem 3 to low-rank plus diagonal tensor approximation, achieving  $k' = O(k^2/\epsilon^4)$  and the low-rank plus sparse tensor approximation problem achieving  $k' = O\left(\frac{k^2 \cdot t^4}{\epsilon^6}\right)$ , where  $t$  is the maximum number of zeros on any face of  $W$ .

We also consider a common variant of low-rank approximation studied in data mining and information retrieval: *Boolean low-rank approximation* (binary low-rank approximation). Here one is given binary  $A \in \{0, 1\}^{n \times n}$  and seeks to find  $U \in \{0, 1\}^{n \times k}$  and  $V \in \{0, 1\}^{k \times n}$  minimizing  $\|A - U \cdot V\|_0$  where  $U \cdot V$  denotes Boolean matrix multiplication and  $\|\cdot\|_0$  is the entrywise  $\ell_0$  norm, equal to the squared Frobenius norm in this case. While Boolean low-rank approximation is NP-hard [20, 26], there is a large body of work studying heuristic algorithms and approximation schemes, when no entries of  $A$  are masked [46, 59, 67, 10, 24]. We show in the full paper that any black-box algorithm for standard Boolean low-rank approximation yields a bicriteria algorithm for masked Boolean low-rank approximation, with rank depending on the *nondeterministic communication complexity* of the mask  $W$ .

► **Theorem 4** (Nondeterministic Communication Complexity  $\rightarrow$  Boolean Low-Rank Approx).

Let  $f$  be the function computed by  $W$  and  $N(f)$  be the nondeterministic communication complexity of  $f$ . For any  $k' \geq k \cdot 2^{N(f)}$ , if one computes  $U, V \in \{0, 1\}^{n \times k'}$  satisfying  $\|A \circ W - U \cdot V\|_0 \leq \min_{\hat{U}, \hat{V} \in \{0, 1\}^{n \times k'}} \|A \circ W - \hat{U} \cdot \hat{V}\|_0 + \Delta$  then:

$$\|W \circ (A - U \cdot V)\|_0 \leq 2^{N(f)} \cdot OPT + \Delta,$$

where  $OPT = \min_{\hat{U}, \hat{V} \in \{0, 1\}^{k \times n}} \|W \circ (A - \hat{U} \cdot \hat{V})\|_0$  and  $U \cdot V$  denotes Boolean matrix multiplication.

We can apply Theorem 4 for example, to the low-rank plus diagonal Boolean matrix approximation problem, where  $W$  is zero on its diagonal and one everywhere else. In this case we have  $2^{N(f)} = \log n$  and correspondingly  $k' = k \log n$ .

## 1.2.4 Connections to Approximate Rank and Other Communication Lower Bounds

In Section 1.3 we sketch the proof of Theorem 1, which is very simple (Theorems 2, 3, and 4 are proved similarly.) The proof is based on covering  $W$  with  $2^{R_{\epsilon}^{1-sided}(\neg f)}$  disjoint monochromatic rectangles, which match  $W$  on all but a small random subset of its 1 entries. The existence of a 1-sided error randomized communication protocol for  $\neg f$  using  $R_{\epsilon}^{1-sided}(\neg f)$  bits of communication is well known to imply the existence of such a covering with  $2^{R_{\epsilon}^{1-sided}(\neg f)}$  rectangles. However, the optimal size of such a covering, which is known as the “public-coin partition bound” [34], may be lower than this. In fact, recent work has shown that it is provably smaller for some problems [27]. Thus, our algorithm can be stated in terms of this bound, giving improved results for these problems. However, as far as we are aware, this bound does not give any improvements for the communication problems we consider (corresponding to natural weight matrices  $W$ ).

The public coin partition bound is a strengthening of the well-studied partition bound [33] for randomized communication complexity, which is itself a strengthening of the smooth rectangle bound [33]. This logarithm of the smooth rectangle bound is equivalent to the log approximate nonnegative rank of  $W$  up to constants [38]. It has been shown that the randomized communication complexity can be polynomially larger than the log partition bound [27]. Additionally, recent work refuting the log approximate rank conjecture [16] has shown that the randomized communication complexity can be exponentially larger than the log approximate nonnegative rank. Thus, improving our results to depend on these communication complexity lower bounds rather than the communication complexity itself would lead to potential improvements for some weight matrices  $W$ . However, all known separations are for  $W$  with complex structure and relatively high communication complexity, and thus not relevant to common applications. Additionally, it is unclear how to extend

our techniques to these weaker notions, or to other related notations, such as information complexity [14]. Such extensions would be interesting, e.g., connecting the difficulty of masked low-rank approximation to the approximate rank of the mask.

### 1.2.5 Lower Bounds

Given our results, and the above discussion, a natural question to ask is:

*Is there a natural notion of the complexity of the mask  $W$  that characterizes the difficulty of the masked low-rank approximation problem?*

We give some initial results, focused on how communication complexity in particular relates to the best bicriteria approximation factor for masked low-rank approximation achievable in polynomial time. We note that, since our results actually hold with rank depending on the public-coin partition bound [34], which has been separated from the randomized communication complexity, the communication complexity itself certainly does not tightly characterize the difficulty of masked low-rank approximation. However, we view our lower bounds in terms of communication complexity as a step in understanding this difficulty.

We prove two bounds based on a conjecture of the hardness of approximate 3-coloring. We show that there is a class of masks  $W$  such that any polynomial time algorithm achieving guarantee (1) and small enough  $\epsilon$  requires bicriteria rank  $k' = \Omega\left(\frac{D(f)}{\log D(f)}\right)$  where  $D(f)$  is the deterministic communication complexity of  $f$ . Note that  $D(f)$  is only greater than  $R_\epsilon^{1-sided}(\neg f)$  and  $R_\epsilon(f)$ .

We strengthen this bound significantly for two natural variants of the masked low-rank approximation problem: when the low-rank approximation  $L$  is required to have a non-negative or binary factorization. We note that our techniques yield matching algorithmic results analogous to Theorems 1 and 2 for these variants. We show that for these variants on Problem 1, there is a class of masks  $W$  such that any polynomial time algorithm achieving guarantee (1) for small enough  $\epsilon$  requires bicriteria rank which is exponential in the deterministic communication complexity,  $k' = 2^{\Omega(D(\neg f))}$ . This bound matches our algorithmic results for these variants. We note that in the parameter regimes considered (we just require rank  $k = 3$ ), there exist polynomial time algorithms for the *non-masked* versions of binary and non-negative low-rank approximation. Thus, the hardness in terms of communication complexity comes from adding the mask to the low-rank cost function rather than the binary and non-negativity constraints themselves.

Our lower bounds are closely related to those of [30] on the hardness of bicriteria low-rank matrix completion. We note that for any  $n \times n$  mask matrix  $W$ , we can always bound  $D(f) = O(\log n)$ . Thus, achieving a  $2^{o(D(f))}$  bicriteria approximation factor means achieving an approximation factor sub-polynomial in  $n$ . [30] leaves open if achieving a  $\sqrt{n}$  bicriteria approximation to rank-3 matrix completion is hard (Question 4.3 in [30]), and more generally asks what bicriteria approximation is achievable in polynomial time (Question 4.2 in [30]).

## 1.3 Our Techniques

The key ideas behind Theorems 1 and 2 are similar. We focus on Theorem 1 for exposition. We want to argue that any near optimal rank- $k'$  approximation of  $A \circ W$ , gives a good bicriteria solution to the masked rank- $k$  approximation problem. For simplicity, here we focus on showing this for the actual optimal rank- $k'$  approximation,  $L = \arg \min_{\text{rank} = k'} \hat{L} \|(A \circ W) - \hat{L}\|_F^2$ . We show that  $\|W \circ (A - L)\|_F^2 \leq OPT + O(\epsilon) \|A \circ W\|_F^2$  via a comparison method. Namely, we exhibit a rank  $k'$  matrix  $\tilde{L}$  that:



1. Nearly matches how well the optimum rank- $k$  solution  $L_{opt}$  to Problem 1 approximates  $A$  on the support of  $W$ . In particular,  $\|(A - \bar{L}) \circ W\|_F^2 \leq \|(A - L_{opt}) \circ W\|_F^2 + O(\epsilon) \|A \circ W\|_F^2$ .
2. Places *no mass* outside the support of  $W$ . In particular,  $\|\bar{L} \circ (1 - W)\|_F^2 = 0$ .

Since  $L$  minimizes the distance to  $(A \circ W)$  among all rank- $k'$  matrices, we have  $\|(A \circ W) - L\|_F^2 \leq \|(A \circ W) - \bar{L}\|_F^2$ . However, by (2),  $\bar{L}$  *exactly matches*  $A \circ W$  outside the support of  $W$  – both matrices are 0 there. Thus  $L$  must have at least as large error outside the support of  $W$ , and in turn cannot have larger error on the support of  $W$ . That is, we must have  $\|(A - L) \circ W\|_F^2 \leq \|(A - \bar{L}) \circ W\|_F^2$ . Then by (1),  $L$  satisfies the desired bound.

### 1.3.1 From Communication Protocols to Low-Rank Approximations

The key question becomes how to exhibit  $\bar{L}$ , which we do using communication complexity. We view  $W$  as the communication matrix of some function  $f : \{0, 1\}^{\log n} \times \{0, 1\}^{\log n} \rightarrow \{0, 1\}$ , with  $W_{x,y} = f(x, y)$ , where in  $f$  we interpret  $x, y \in [n]$  as their binary representations. It is well-known that the existence of a deterministic communication protocol  $\Pi$  that computes  $f$  with  $D(f)$  total bits of communication implies the existence of a partition of  $W$  into  $2^{D(f)}$  *monochromatic combinatorial rectangles*. That is, there are  $2^{D(f)}$  non-overlapping sets  $R_i = S \times T$  for  $S, T \in [n]$  that partition  $W$  and that satisfy  $W(R_i)$  is either all 1 or all 0. We could construct  $\bar{L}$  by taking the best  $k$ -rank approximation of each  $A(R_i)$  where  $R_i$  is colored 1 (i.e., contains inputs with  $f(x, y) = 1$ ). We could then sum up these approximations to produce  $\bar{L}$  with rank  $\leq k \cdot 2^{D(f)}$ . Note that  $\bar{L}$  is 0 outside the rectangles colored 1 – i.e., outside the support of  $W$ . Thus condition (2) above is satisfied. Further,  $\bar{L}$  matches the optimal rank- $k$  approximation on each  $R_i$  colored 1. So it approximates  $A$  at least as well as  $L_{opt}$  on these rectangles, and since these rectangles fully cover the support of  $W$  we have  $\|(A - \bar{L}) \circ W\|_F^2 \leq \|(A - L_{opt}) \circ W\|_F^2$ , giving the requirement of (1).

Unfortunately, essentially none of the  $W$  that are of interest in applications admit efficient deterministic communication protocols.  $k' = k \cdot 2^{D(f)}$  will typically be larger than  $n$ , giving a vacuous bound. Thus we turn to randomized communication complexity with error probability  $\epsilon$ ,  $R_\epsilon(f)$ , which is much lower in these cases. A randomized protocol  $\Pi$  achieving this complexity corresponds to a distribution over partitions of  $W$  into  $2^{R_\epsilon(f)}$  rectangles. These rectangles are not monochromatic but are close to it – letting  $W_\Pi$  be the communication matrix of the (random) function computed by the protocol,  $W_\Pi$  is partitioned into  $2^{R_\epsilon(f)}$  monochromatic rectangles and further matches  $W$  on each  $(x, y)$  with probability at least  $1 - \epsilon$ . We prove that, even with this small error, constructing  $\bar{L}$  as above using the partition of  $W_\Pi$  instead of  $W$  itself gives a solution nearly matching  $L_{opt}$  up to small additive error. This error will involve  $\|A \circ W\|_F^2$  and  $\|L_{opt} \circ (1 - W)\|_F^2$ , depending on whether the protocol makes 1 or 2-sided error, as seen in Theorems 1 and 2.

### 1.3.2 Low-Rank Approximation to $W$ Does Not Suffice

A natural view of our argument above is that the existence of an efficient randomized protocol for  $W$  implies the existence of a distribution over low-rank matrices (induced by partitions into near monochromatic rectangles) that match  $W$  on each entry with good probability. We note that this distributional view is critical – simply having a low-rank approximation to  $W$  matching all but a small fraction of entries does not suffice. The mistaken entries could in the worst case align with very heavy entries of  $A$ , which must be approximated well to solve masked low-rank approximation to small error. An approximation with small entrywise error (in the  $\ell_\infty$  sense) would suffice. However, for important cases, e.g., when  $W$  is zero on the diagonal and one off the diagonal, such approximations provably require higher rank than  $2^{R_\epsilon(f)}$  and thus relying on them would lead to significantly weaker bounds [2].

### 1.3.3 Other Communication Models

In extending our results to other communication models, we first consider the connection between multiparty number-in-hand communication and tensor low-rank approximation. Protocols in this model correspond to a partition of the communication tensor  $W \in \{0, 1\}^{n \times n \times n}$  into  $2^{R_\epsilon^3(f)}$  monochromatic (or nearly monochromatic) rectangles of the form  $R_i = S \times T \times U$  for  $S, T, U \subseteq [n]$ , where  $R_\epsilon^3(f)$  is the randomized 3-player communication complexity of  $W$ . We can again argue the existence of a rank  $k' = k \cdot 2^{R_\epsilon^3(f)}$  tensor  $\bar{L}$ , obtained by taking a near optimal low-rank approximation to each rectangle colored 1 in  $W_\Pi$ , which is mostly 0 outside the support of  $W$  and at the same time competes with the best rank- $k$  tensor approximation  $L_{opt}$  on the support of  $W$ . There are different notions of rank for tensors; here we mostly discuss canonical or CP rank. This lets us argue, as in the two player case, that the best rank- $k'$  approximation of  $A \circ W$  also competes with  $L_{opt}$ . It is not known how to find this best rank- $k'$  approximation efficiently, however using an algorithm of [63] we can find a rank  $k'' = O((k'/\epsilon)^2)$  bicriteria approximation achieving relative error  $1 + \epsilon$ . Overall we have  $k'' = O\left((k/\epsilon)^2 \cdot 2^{2R_\epsilon^3(f)}\right)$ , giving Theorem 3.

We next consider the nondeterministic communication complexity. In a nondeterministic communication protocol for a function  $f$ , players can make “guesses” at any point during the protocol  $\Pi$ . The only requirement is that, (1) for every  $x, y$  with  $f(x, y) = 1$ , for some set of guesses made by the players, the protocol outputs  $\Pi(x, y) = 1$  and (2) the protocol never outputs  $\Pi(x, y) = 1$  for  $x, y$  with  $f(x, y) = 0$ . Such a protocol using  $N(f)$  bits of communication corresponds to covering the communication matrix  $W$  with  $2^{N(f)}$  *possibly overlapping* monochromatic rectangles. In many cases, the nondeterministic complexity is much lower than the randomized communication complexity. However, for low-rank approximation in the Frobenius norm, the overlap is a problem. We cannot construct  $\bar{L}$  simply by approximating each rectangle and adding these approximations together.  $\bar{L}$  will be too “heavy” where the rectangles overlap. However, for the Boolean low-rank approximation problem, the overlap is less of a problem. We simply construct  $\bar{L}$  in the same way, letting it be the AND of the approximations on each rectangle. In the end, we obtain an error bound of roughly  $2^{N(f)} \cdot OPT$ , owing to the fact that error may still build up on the overlapping sections. Since there are  $2^{N(f)}$  rectangles total, each entry is overlapped by at most  $2^{N(f)}$  of them. However, since  $N(f)$  can be very small, this result gives a tradeoff with Theorems 1 and 2 (which can also be extended to the Boolean case). For example, we show how to obtain error  $\approx O(\log n \cdot OPT)$  for the Boolean low-rank plus diagonal approximation problem, with rank  $k' = O(k \log n)$ . This is smaller than the  $O(k/\epsilon)$  achieved by Theorem 1 for small  $\epsilon$ , which may be required to achieve good error if, e.g.,  $\|A\|_F^2$  is large.

### 1.3.4 An Alternative Approach

In the important cases when  $W$  is zero on its diagonal and one elsewhere or has a few non-zeros per row (the low-rank plus diagonal and low-rank plus sparse approximation problems, respectively) the existence of  $\bar{L}$  satisfying the necessary conditions (1) and (2) above can be proven via a very different technique. The key idea is a structural result: that any low-rank matrix cannot concentrate too much weight on more than a few entries of its diagonal, or more generally, on a sparse support outside a few rows. Thus we can obtain  $\bar{L}$  from  $L_{opt}$  by explicitly zero-ing out these few large entries falling outside the support of  $W$  (e.g., on its diagonal when  $W$  has zeros just on its diagonal). We detail this approach in the full paper, giving a bound matching Theorem 1 in this case. We show that the same structural result can also be used to obtain a fixed-parameter-tractable, relative error, non-bicriteria



approximation algorithm for Problem 1 in the low-rank plus diagonal case, as well as for the closely related factor analysis problem. We are unaware of any formal connection between this structural result and our communication complexity framework; however, establishing one would be very interesting.

## 1.4 Road Map

In Section 2 we give preliminaries, defining the communication models we use and giving communication complexity bounds for common mask matrices in these models. In Section 3 we then prove our main results, Theorems 1 and 2. We instantiate these results for the common mask matrices shown in Table 1. We defer our results connecting masked tensor approximation to multiparty communication complexity and Boolean low-rank approximation to nondeterministic communication complexity to the full paper – available at <https://arxiv.org/abs/1904.09841>. We also defer our lower bound results to the full version.

## 2 Preliminaries

### 2.1 Notation and Conventions

Throughout we use  $\log z$  to denote the base-2 logarithm of  $z$ . For simplicity, so that we can associate any  $W \in \mathbb{R}^{n \times n}$  with a function  $f : \{0, 1\}^{\log n} \times \{0, 1\}^{\log n} \rightarrow \{0, 1\}$  we assume that  $n$  is a power of 2 and so  $\log n$  is an integer. Our results can be easily extended to general  $n$ . Given a matrix  $M \in \mathbb{R}^{n \times n}$  and a combinatorial rectangle  $R = S \times T$  for  $S, T \subseteq [n]$ , we let  $M_R$  denote the submatrix of  $M$  indexed by  $R$ . For matrix  $M$  we let  $1 - M$  denote the matrix  $N$  with  $N_{i,j} = 1 - M_{i,j}$ . E.g.,  $1 - I$  is the matrix with all zeros on diagonal and all ones off diagonal.

While in the introduction we focus on low-rank approximation in the Frobenius norm, many of our results will apply to any entrywise matrix norm of the form:

► **Definition 5.** An entrywise matrix norm  $\|\cdot\|_* : \mathbb{R}^{n \times n} \rightarrow \mathbb{R}$  is a function of the form:

$$\|M\|_* = \sum_{i=1}^n \sum_{j=1}^n g(|M_{i,j}|),$$

where  $g : \mathbb{R} \rightarrow \mathbb{R}$  is some monotonically increasing nonnegative function.

$g(x) = x^2$  gives the squared Frobenius norm,  $g(x) = x^p$  gives the entrywise  $\ell_p$  norm,  $g(x) = 1$  iff  $x \neq 0$  gives the entrywise  $\ell_0$  norm, etc. See [62, 10, 17, 5] for a discussion of standard low-rank approximation algorithms for these norms. As discussed, our bicriteria results will simply require applying one of these algorithms to compute a near-optimal low-rank approximation to  $A \circ W$  (i.e.,  $A$  with the masked entries zeroed out).

### 2.2 Communication Complexity Models

We give a brief introduction to the communication models we consider, and refer the reader to the textbooks [39, 53] for more background. We mostly consider two-party communication of Boolean functions, though in the full paper discuss extensions to more than two parties.

Consider two parties, Alice and Bob, holding inputs  $x \in \mathcal{X}$  and  $y \in \mathcal{Y}$  respectively. They exchange messages in order to compute a function  $f : \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$  evaluated at  $(x, y)$ . They would like to do this while minimizing the total number of bits exchanged. The communication between the parties is determined by a possibly randomized protocol, which

specifies the message of the next player to speak as a function of previous messages received by that player and that player's input. For a given protocol  $\Pi$ , we let  $|\Pi(x, y)|$  denote the number of bits transmitted by the players on inputs  $x$  and  $y$ , and we let  $|\Pi| = \max_{x, y} |\Pi(x, y)|$ .

Let  $M$  be the communication matrix of  $f$ , that is, the matrix whose rows are indexed by elements of  $\mathcal{X}$  and columns by elements of  $\mathcal{Y}$ , and for which  $M_{x, y} = f(x, y)$ . A well known and useful property is that  $\Pi$  partitions  $M$  into rectangles  $R = S \times T$ , where  $S \subseteq \mathcal{X}$  and  $T \subseteq \mathcal{Y}$ , and every pair  $(x, y)$  of inputs with  $(x, y) \in S \times T$  has the same output when running protocol  $\Pi$ . The number of rectangles in the partition is equal to  $2^{|\Pi|}$ . We call the unique output of  $\Pi$  on a rectangle  $S \times T$  the *label* of the rectangle.

► **Definition 6** (Deterministic Communication Complexity). *The deterministic communication complexity  $D(f) = \min_{\Pi} |\Pi|$ , where the minimum is taken over all protocols  $\Pi$  for which  $\Pi(x, y) = f(x, y)$  for every pair  $(x, y)$  of inputs. Equivalently,  $D(f)$  is the minimum number so that  $M$  can be partitioned via a protocol  $\Pi$  into  $2^{D(f)}$  rectangles for which for every rectangle  $R$  and  $b \in \{0, 1\}$ , if  $R$  is labeled  $b$ , then for all  $(x, y) \in R$ ,  $f(x, y) = b$ .*

We next turn to randomized communication complexity. For the purposes of this paper, we will consider public coin randomized communication complexity, i.e., there is a shared random string  $r$  that both Alice and Bob have access to. In a randomized protocol  $\Pi$ , Alice and Bob see  $r$  and then run a deterministic protocol  $\Pi_r$ . We say a protocol  $\Pi$  is a  $(\delta_1, \delta_2)$ -error protocol if for all  $x, y \in \mathcal{X} \times \mathcal{Y}$ , with  $f(x, y) = 1$ ,  $\mathbb{P}_r[\Pi_r(x, y) = f(x, y)] \geq 1 - \delta_1$  and for all  $x, y \in \mathcal{X} \times \mathcal{Y}$  with  $f(x, y) = 0$ ,  $\mathbb{P}_r[\Pi_r(x, y) = f(x, y)] \geq 1 - \delta_2$ . We can then define a general notion of randomized communication complexity:

► **Definition 7** (Randomized Communication Complexity – General). *The  $(\delta_1, \delta_2)$ -error randomized communication complexity  $R_{\delta_1, \delta_2}(f) = \min_{\Pi} |\Pi|$ , where the minimum is taken over all  $(\delta_1, \delta_2)$ -error protocols  $\Pi$ . Equivalently,  $R_{\delta_1, \delta_2}(f)$  is the minimum number so that there is a distribution over protocols inducing partitions of  $M$ , each containing at most  $2^{R_{\delta_1, \delta_2}(f)}$  rectangles, such that (1) for every  $(x, y) \in \mathcal{X} \times \mathcal{Y}$  with  $f(x, y) = 1$ , with probability at least  $1 - \delta_1$ ,  $(x, y)$  lands in a rectangle which is labeled 1 and (2) for every  $(x, y) \in \mathcal{X} \times \mathcal{Y}$  with  $f(x, y) = 0$ , with probability at least  $1 - \delta_2$ ,  $(x, y)$  lands in a rectangle which is labeled 0.*

Definition 7 is typically specialized to two cases: the randomized communication complexity with 2-sided error and the randomized communication complexity with 1-sided error.

► **Definition 8** (Randomized Communication Complexity – 2-sided). *The  $\delta$ -error randomized communication complexity of  $f$  is  $R_{\delta}(f) \stackrel{\text{def}}{=} R_{\delta, \delta}(f)$ .*

► **Definition 9** (Randomized Communication Complexity – 1-Sided). *The  $\delta$ -error 1-sided randomized communication complexity of  $f$  is  $R_{\delta}^{1\text{-sided}}(f) \stackrel{\text{def}}{=} R_{0, \delta}(f)$ .*

In Theorem 1 we consider the 1-sided communication complexity of  $\neg f$ :  $R_{\delta, 0}(f)$ .

## 2.3 Specific Communication Bounds

We discuss a few problems that will be particularly useful for our applications. We only need communication upper bounds and in specific models. Note that in this section, as is standard, we state bounds for communication problems with  $n$ -bit inputs. In our applications to masked low-rank approximation, we will typically apply the bounds with input size  $\log n$ .

### Equality

In the Equality problem, denoted  $EQ$ , there are two players Alice and Bob, holding strings  $x, y \in \{0, 1\}^n$ , and the function  $EQ(x, y) = 1$  if  $x = y$ , and  $EQ(x, y) = 0$  otherwise.

► **Theorem 10** ([39], combining Corollaries 26 and 27 of [11]).  $R_\delta^{1-way}(EQ) \leq (1 - \delta) \log((1 - \delta)^2 / \delta) + 5$ , and  $R_\delta^{1-way, 1-sided}(EQ) \leq \log(1/\delta) + 5$ .

We also can bound the nondeterministic communication complexity of inequality, i.e., the function  $NEQ(x, y)$  with  $NEQ(x, y) = 1$  iff  $x \neq y$ .

► **Theorem 11.**  $N(NEQ) \leq \lceil \log n \rceil + 2$ .

**Proof.** Alice simply guesses an index at which  $x$  and  $y$  differ and sends this index (using  $\lceil \log n \rceil$  bits) along with the value of  $x$  at this index to Bob. Bob sends the value of  $y$  at this index and the players check if  $x$  and  $y$  differ at the index. ◀

Essentially the same protocol can be used to solve the negation of the disjointness problem, with  $\neg DISJ(x, y) = 1$  only if there is some  $k \in [n]$  with  $x(k) = y(k) = 1$ . We thus have:

► **Theorem 12** ([60]).  $N(\neg DISJ) \leq \lceil \log n \rceil + 2$ .

### Greater-Than

In Greater-Than, denoted  $GT$ , there are two players Alice and Bob, holding integers  $x, y \in \{0, 1, \dots, n-1\}$ , and the function  $GT(x, y) = 1$  if  $x > y$ , and  $GT(x, y) = 0$  otherwise.

► **Theorem 13** ([52]).  $R_\delta(GT) = O(\log(n/\delta))$ .

## 3 Bicriteria Approximation from Communication Complexity

In this section we prove our main results, Theorems 1 and 2, which connect the randomized communication complexity of the binary matrix  $W$  to the rank required to solve Problem 1 efficiently up to small additive error. We prove a general theorem connecting the rank to  $R_{\delta_1, \delta_2}(f)$ . Both Theorems 1 and 2 follow as corollaries if we consider the 1-sided error complexity  $R_\delta^{1-sided}(\neg f) = R_{\delta, 0}(f)$  and the 2-sided error complexity  $R_\delta(f) \stackrel{\text{def}}{=} R_{\delta, \delta}(f)$  respectively (Definitions 8 and 9).

► **Theorem 14** (Randomized Communication Complexity  $\rightarrow$  Bicriteria Approximation). *Consider  $W \in \{0, 1\}^{n \times n}$  and let  $f$  be the function computed by it. For  $k' \geq k \cdot 2^{R_{\epsilon_1, \epsilon_2}(f)}$ , and any entrywise norm  $\|\cdot\|_\star$  (Def. 5), for any  $L$  satisfying  $\|A \circ W - L\|_\star \leq \min_{\text{rank}-k'} \hat{L} \|A \circ W - \hat{L}\|_\star + \Delta$ :*

$$\|(A - L) \circ W\|_\star \leq OPT + \epsilon_1 \|A \circ W\|_\star + \epsilon_2 \|L_{opt} \circ (1 - W)\|_\star + \Delta,$$

where  $OPT = \min_{\text{rank}-k} \hat{L} \|(A - \hat{L}) \circ W\|_\star$  and  $L_{opt}$  is any rank- $k$  matrix achieving  $OPT$ .

**Proof.** As discussed (Def. 7),  $R_{\epsilon_1, \epsilon_2}(f)$  is the minimum number so that there is a distribution on protocols inducing partitions of  $W$ , each containing at most  $2^{R_{\epsilon_1, \epsilon_2}(f)}$  rectangles, such that (1) for every  $x, y \in \{0, 1\}^{\log n}$  with  $f(x, y) = 1$ ,  $(x, y)$  lands in a rectangle labeled 1 with probability  $\geq 1 - \epsilon_1$  and (2) for every  $x, y \in \{0, 1\}^{\log n}$  with  $f(x, y) = 0$ ,  $(x, y)$  lands in a rectangle labeled 0 with probability  $\geq 1 - \epsilon_2$ . In other words, letting  $W_\Pi$  be the (random) matrix corresponding to the function computed by the protocol: (1)  $W \circ (1 - W_\Pi)$  has each entry equal to 1 with probability  $\leq \epsilon_1$  and (2)  $W_\Pi \circ (1 - W)$  has each entry equal to 1 with probability  $\leq \epsilon_2$ . Thus, fixing some  $L_{opt}$ :

$$\begin{aligned} \mathbb{E}_{\text{protocol } \Pi} [\|A \circ W \circ (1 - W_\Pi)\|_* + \|L_{\text{opt}} \circ W_\Pi \circ (1 - W)\|_*] \\ \leq \epsilon_1 \|A \circ W\|_* + \epsilon_2 \|L_{\text{opt}} \circ (1 - W)\|_*. \end{aligned}$$

Thus, there is at least one protocol  $\Pi$  (inducing a partition with  $\leq 2^{R_{\epsilon_1, \epsilon_2}(f)}$  rectangles) with:

$$\|A \circ W \circ (1 - W_\Pi)\|_* + \|L_{\text{opt}} \circ W_\Pi \circ (1 - W)\|_* \leq \epsilon_1 \|A \circ W\|_* + \epsilon_2 \|L_{\text{opt}} \circ (1 - W)\|_*. \quad (2)$$

Let  $P_1$  be the set of rectangles on which the protocol achieving (2) returns 1 and  $P_0$  be the set on which it returns 0. For any  $R \in P_1$  let  $L^R = \arg \min_{\text{rank}-k} \hat{L} \|A_R \circ W_R - \hat{L}\|_*$  (note that  $L^R$  is the size of  $R$ ). Let  $\bar{L}^R$  be the  $n \times n$  matrix equal to  $L^R$  on  $R$  and 0 elsewhere. Let  $\bar{L} = \sum_{R \in P_1} \bar{L}^R$ . Note that  $\bar{L}$  has rank at most  $\sum_{R \in P_1} \text{rank}(\bar{L}^R) \leq k \cdot |P_1| \leq k \cdot 2^{R_{\epsilon_1, \epsilon_2}(f)}$ . Thus, by the assumption that  $L$  satisfies  $\|A \circ W - L\|_* \leq \min_{\text{rank}-k'} \hat{L} \|A \circ W - \hat{L}\|_* + \Delta$ :

$$\begin{aligned} \|(A - L) \circ W\|_* &\leq \|A \circ W - L\|_* \leq \|A \circ W - \bar{L}\|_* + \Delta \\ &= \|(A \circ W - \bar{L}) \circ W_\Pi\|_* + \|(A \circ W - \bar{L}) \circ (1 - W_\Pi)\|_* + \Delta \\ &= \|(A \circ W - \bar{L}) \circ W_\Pi\|_* + \|A \circ W \circ (1 - W_\Pi)\|_* + \Delta, \end{aligned} \quad (3)$$

where the third line follows since  $\bar{L}$  is 0 outside the support of  $W_\Pi$  (i.e., outside of the rectangles in  $P_1$ ). Since  $\bar{L}$  is equal to the best rank- $k$  approximation to  $A_R \circ W_R$  on each rectangle  $R$  in  $P_1$ , and since these rectangles partition the support of  $W_\Pi$ :

$$\begin{aligned} \|(A \circ W - \bar{L}) \circ W_\Pi\|_* &\leq \|(A \circ W - L_{\text{opt}}) \circ W_\Pi\|_* \\ &= \|(A - L_{\text{opt}}) \circ W \circ W_\Pi\|_* + \|L_{\text{opt}} \circ (1 - W) \circ W_\Pi\|_* \\ &\leq \text{OPT} + \|L_{\text{opt}} \circ (1 - W) \circ W_\Pi\|_*. \end{aligned}$$

Plugging back into (3) and applying (2):

$$\begin{aligned} \|(A - L) \circ W\|_* &\leq \text{OPT} + \|L_{\text{opt}} \circ (1 - W) \circ W_\Pi\|_* + \|A \circ W \circ (1 - W_\Pi)\|_* + \Delta \\ &\leq \text{OPT} + \epsilon_1 \|A \circ W\|_* + \epsilon_2 \|L_{\text{opt}} \circ (1 - W)\|_* + \Delta, \end{aligned}$$

which completes the theorem.  $\blacktriangleleft$

**Proof of Theorems 1 and 2.** Theorems 1 and 2 follow by applying Theorem 14 with  $\epsilon_1 = \epsilon_2 = \epsilon$  and  $\epsilon_1 = \epsilon$ ,  $\epsilon_2 = 0$  respectively, and noting that  $\|A \circ W\|_* \leq \|A\|_*$  and  $\|L_{\text{opt}} \circ (1 - W)\|_* \leq \|L_{\text{opt}}\|_*$ . When  $\|\cdot\|_*$  is the squared Frobenius norm,  $L$  satisfying  $\|A \circ W - L\|_* \leq \min_{\text{rank}-k'} \hat{L} \|A \circ W - \hat{L}\|_* + \Delta$  for  $\Delta = \epsilon \|(A \circ W) - (A \circ W)_{k'}\|_F^2 \leq \epsilon \|A \circ W\|_F^2$  can be computed with high probability in  $O(\text{nnz}(A)) + n \cdot \text{poly}(k'/\epsilon)$  time.  $\blacktriangleleft$

### 3.1 Applications of Main Theorem

We can instantiate Theorem 14 for a number of common mask patterns, yielding the results summarized in Table 1. Note that the additive error bounds achieved are stated in terms of  $\|A \circ W\|_*$  and  $\|L_{\text{opt}} \circ (1 - W)\|_*$ , which are only smaller than  $\|A\|_*$  and  $\|L_{\text{opt}}\|_*$  respectively.

We start with the case when  $W$  is the negation of a diagonal matrix or a block diagonal matrix, corresponding to the Low-Rank Plus Diagonal (LRPD) and Low-Rank Plus Block Diagonal (LRPBD) matrix approximation problems. The argument uses the communication complexity of Equality (EQ). A variant on this problem is also used when  $W$  has at most  $t$  nonzeros (or nonzero blocks) per row. This corresponds to the Low-Rank Plus Sparse (LRPS) and Low-Rank Plus Block Sparse (LRPBS) approximation problems, which strictly generalize the Low-Rank Plus (Block) Diagonal Problem. Proofs are given in the full paper.

► **Corollary 15** (Low-Rank Plus Diagonal Approximation). *Let  $W = 1 - I$  where  $I$  is the  $n \times n$  identity matrix. Then for  $k' = O\left(\frac{k}{\epsilon}\right)$  and  $L$  with  $\|A \circ W - L\|_* \leq \min_{\text{rank} - k'} \hat{L} \|A \circ W - \hat{L}\|_* + \epsilon \|A \circ W\|_*$ :*

$$\|(A - L) \circ W\|_* \leq OPT + 2\epsilon \|A \circ W\|_*.$$

*If  $\|\cdot\|_* = \|\cdot\|_F^2$ ,  $L$  can be computed with high probability in  $O(\text{nnz}(A)) + n \text{poly}(k/\epsilon)$  time.*

**Proof.** The function  $f$  corresponding to  $W$  is the inequality function  $NEQ$ . We have  $R_\epsilon^{1\text{-sided}}(\neg NEQ) = R_\epsilon^{1\text{-sided}}(EQ)$ , which by Theorem 10 is bounded by  $\log(1/\epsilon) + 5$ . Thus  $2^{R_\epsilon^{1\text{-sided}}(\neg NEQ)} \leq \frac{32}{\epsilon}$ . The corollary then follows directly from Theorem 14. ◀

► **Corollary 16** (Low-Rank Plus Block Diagonal Approximation). *Consider any partition  $B_1 \cup B_2 \cup \dots \cup B_b = [n]$  and let  $W$  be the matrix with  $W_{i,j} = 0$  if  $i, j \in B_k$  for some  $k$  and  $W_{i,j} = 1$  otherwise. Then for  $k' = O\left(\frac{k}{\epsilon}\right)$  and  $L$  with  $\|A \circ W - L\|_* \leq \min_{\text{rank} - k'} \hat{L} \|A \circ W - \hat{L}\|_* + \epsilon \|A \circ W\|_*$ :*

$$\|(A - L) \circ W\|_* \leq OPT + 2\epsilon \|A \circ W\|_*.$$

*If  $\|\cdot\|_* = \|\cdot\|_F^2$ ,  $L$  can be computed with high probability in  $O(\text{nnz}(A)) + n \text{poly}(k/\epsilon)$  time.*

**Proof.** The function  $f$  corresponding to  $W$  is the inequality function  $NEQ$  where  $x, y \in [n]$  are identified with  $j, k \in [b]$  if block  $B_j$  contains  $x$  and  $B_k$  contains  $y$ . The randomized communication complexity  $\neg f$  is thus bounded by the complexity of equality. By Theorem 10,  $R_\epsilon^{1\text{-sided}}(EQ) \leq \log(1/\epsilon) + 5$  and so  $2^{R_\epsilon^{1\text{-sided}}(f)} \leq \frac{32}{\epsilon}$ , which gives the corollary. ◀

Beyond equality, a number of common sparsity patterns are related to the communication complexity of Greater-Than (GT), which is bounded by Theorem 13. Since two-sided error is required to give efficient GT protocols, we incur an additional error term depending on  $L_{opt}$ . An interesting question is if this is necessary for efficient bicriteria approximation.

► **Corollary 17** (Low-Rank Plus Banded Approximation). *For any integer  $p \leq n$ , let  $W \in \{0, 1\}^{n \times n}$  be the banded Toeplitz matrix with  $W_{i,j} = 0$  iff  $|i - j| < p$ . Then for  $k' = k \cdot \min\left(\frac{p}{\epsilon}, \text{poly}\left(\frac{\log n}{\epsilon}\right)\right)$  and  $L$  with  $\|A \circ W - L\|_* \leq \min_{\text{rank} - k'} \hat{L} \|A \circ W - \hat{L}\|_* + \epsilon \|A \circ W\|_*$ :*

$$\|(A - L) \circ W\|_* \leq OPT + 2\epsilon \|A \circ W\|_* + \epsilon \|L_{opt} \circ (1 - W)\|_*.$$

*If  $\|\cdot\|_* = \|\cdot\|_F^2$ ,  $L$  can be computed with high probability in  $O(\text{nnz}(A)) + n \text{poly}(k'/\epsilon)$  time.*

**Proof.** The function  $f$  corresponding to  $W$  is the negation of the AND of  $i + p < j$  and  $j + p > i$ . Thus, it can be solved with two calls to a protocol for Greater-Than (GT). By Theorem 13, for  $\log n$  bit inputs,  $R_\epsilon(GT) = O\left(\log\left(\frac{\log n}{\epsilon}\right)\right)$ . Thus  $R_\epsilon(f) = O\left(\log\left(\frac{\log n}{\epsilon}\right)\right)$  and  $2^{R_\epsilon(f)} = \text{poly}\left(\frac{\log n}{\epsilon}\right)$ , giving  $k' = k \cdot \text{poly}\left(\frac{\log n}{\epsilon}\right)$ . When  $p$  is small, we can apply our result for  $W$  with sparse rows (see full paper), which gives  $k' = k \cdot \frac{p}{\epsilon}$ , completing the corollary. ◀

In the full paper, we also consider a “multi-dimensional” banded pattern. Here each  $i \in \{0, 1\}^{\log n}$  corresponds to a point  $(i_1, i_2)$  in a  $\sqrt{n} \times \sqrt{n}$  grid ( $i_1$  and  $i_2$  are determined by the first  $\frac{\log n}{2}$  and last  $\frac{\log n}{2}$  bits of  $i$  respectively). We focus on the two-dimensional case, achieving rank  $k' = k \cdot \text{poly}\left(\frac{\log n}{\epsilon}\right)$  as in the 1-dimensional case. This set up can easily be generalized to higher dimensions. We can also imagine generalizing to different distance measures over the points  $(i_1, i_2)$  using efficient sketching methods (which yield efficient communication protocols) for various distances [6, 35].

A similar result holds for low-rank approximation with monotone missing data.

► **Corollary 18** (Monotone Missing Data Problem (MMDP)). *Let  $W \in \{0, 1\}^{n \times n}$  be any matrix where each row of  $W$  has a prefix of an arbitrary number of ones, followed by a suffix of zeros. Then for  $k' = k \cdot \text{poly}\left(\frac{\log n}{\epsilon}\right)$  and  $L$  with  $\|A \circ W - L\|_* \leq \min_{\text{rank} - k'} \hat{L} \|A \circ W - \hat{L}\|_* + \epsilon \|A \circ W\|_*$ :*

$$\|(A - L) \circ W\|_* \leq OPT + 2\epsilon \|A \circ W\|_* + \epsilon \|L_{opt} \circ (1 - W)\|_*.$$

If  $\|\cdot\|_* = \|\cdot\|_F^2$ ,  $L$  can be computed with high probability in  $O(\text{nnz}(A)) + n \text{poly}(k'/\epsilon)$  time.

**Proof.** Let  $p_x$  be the length of the prefix of ones in the  $x^{\text{th}}$  row of  $W$ . Then the function  $f$  corresponding to  $W$  is  $f(x, y) = 1$  iff  $p_x \geq y$ . That is, it is just the Greater-Than function where Alice maps her input  $x$  to  $p_x$ . Thus by Theorem 13,  $R_\epsilon(f) \leq R_\epsilon(GT) = O\left(\log\left(\frac{\log n}{\epsilon}\right)\right)$ . So  $2^{R_\epsilon(f)} = \text{poly}\left(\frac{\log n}{\epsilon}\right)$ , which gives the corollary. ◀

## 4 Open Questions

By focusing on bicriteria approximation, we show how to solve masked low-rank approximation in polynomial time using a simple heuristic. A number of open questions remain. It would be very interesting to improve the bicriteria ranks we achieve for common masks (summarized in Table 1). It would also be interesting to give relative error bounds achieving error  $(1 + \epsilon) \cdot OPT$  instead of our additive error bounds. This is challenging since it requires achieving zero error when there is an exact masked low-rank factorization of  $A$ .

Relatedly, while we have connected bicriteria masked low-rank approximation to the randomized communication complexity of the mask matrix  $W$  (in fact, the public coin partition number of  $W$ ), it would be very interesting to find a notion of  $W$ 's complexity that tightly characterizes the bicriteria rank achievable in polynomial time. We make some initial steps via lower bounds in terms of the communication complexity in the full paper, however the question remains mostly unanswered.

Finally, a related problem is *weighted low-rank approximation* – when  $W$  is real valued and we seek to minimize  $\|W \circ (A - L)\|_F^2$ . Approximation algorithms depending exponentially on the rank  $k$ , error parameter  $\epsilon$ , and notions of  $W$ 's complexity, such as its rank or number of distinct columns are known [54]. However, it would be very interesting to give polynomial time bicriteria approximation algorithms as we have done in the special case of binary  $W$ .

---

## References

- 1 SAS/STAT(R) 9.22 User's guide. See Figure 54.7. URL: [https://support.sas.com/documentation/cdl/en/statug/63347/HTML/defaultviewer.htm#statug\\_mi\\_sect017.htm](https://support.sas.com/documentation/cdl/en/statug/63347/HTML/defaultviewer.htm#statug_mi_sect017.htm).
- 2 Noga Alon. Perturbed identity matrices have high rank: Proof and applications. *Combinatorics, Probability & Computing*, 18(1-2):3, 2009.
- 3 Yossi Azar, Amos Fiat, Anna Karlin, Frank McSherry, and Jared Saia. Spectral analysis of data. In *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing (STOC)*, 2001.
- 4 Masoumeh Azghani and Sumei Sun. Low-rank block sparse decomposition algorithm for anomaly detection in networks. In *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*, pages 807–810. IEEE, 2015.
- 5 Frank Ban, Vijay Bhattiprolu, Karl Bringmann, Pavel Kolev, Euiwoong Lee, and David P Woodruff. A PTAS for  $\ell_p$ -low rank approximation. In *Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 747–766, 2019.



- 6 Ziv Bar-Yossef, TS Jayram, Robert Krauthgamer, and Ravi Kumar. Approximating edit distance efficiently. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 550–559, 2004.
- 7 S. Beghelli, R.P. Guidorzi, and U. Soverini. The Frisch scheme in dynamic system identification. *Automatica*, 26(1):171–176, 1990.
- 8 Peter Benner, Venera Khoromskaia, and Boris N Khoromskij. A reduced basis approach for calculation of the Bethe–Salpeter excitation energies by using low-rank tensor factorisations. *Molecular Physics*, 114(7-8):1148–1161, 2016.
- 9 Aditya Bhaskara, Silvio Lattanzi, Sergei Vassilvitskii, and Morteza Zadimoghaddam. Residual based sampling for online low rank approximation. In *Proceedings of the 60th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 1596–1614, 2019.
- 10 Karl Bringmann, Pavel Kolev, and David Woodruff. Approximation algorithms for  $\ell_0$ -low rank approximation. In *Advances in Neural Information Processing Systems 30 (NeurIPS)*, pages 6648–6659, 2017.
- 11 Joshua Brody, Amit Chakrabarti, Ranganath Kondapally, David P. Woodruff, and Grigory Yaroslavtsev. Certifying equality with limited interaction. *Algorithmica*, 76(3):796–845, 2016.
- 12 Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright. Robust principal component analysis? *Journal of the ACM*, 58(3):11, 2011.
- 13 Emmanuel J Candès and Benjamin Recht. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9(6):717, 2009.
- 14 Amit Chakrabarti, Yaoyun Shi, Anthony Wirth, and Andrew Yao. Informational complexity and the direct sum problem for simultaneous message complexity. In *Proceedings of the 42nd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 270–278, 2001.
- 15 Venkat Chandrasekaran, Sujay Sanghavi, Pablo A Parrilo, and Alan S Willsky. Rank-sparsity incoherence for matrix decomposition. *SIAM Journal on Optimization*, 21(2):572–596, 2011.
- 16 Arkadev Chattopadhyay, Nikhil S Mande, and Suhail Sherif. The log-approximate-rank conjecture is false. In *Proceedings of the 51st Annual ACM Symposium on Theory of Computing (STOC)*, pages 42–53, 2019.
- 17 Flavio Chierichetti, Sreenivas Gollapudi, Ravi Kumar, Silvio Lattanzi, Rina Panigrahy, and David P Woodruff. Algorithms for  $\ell_p$  low-rank approximation. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 806–814, 2017.
- 18 Kenneth L Clarkson and David P Woodruff. Input sparsity and hardness for robust subspace approximation. In *Proceedings of the 56th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 310–329, 2015. [arXiv:1510.06073](#).
- 19 Kenneth L Clarkson and David P Woodruff. Input sparsity and hardness for robust subspace approximation. In *Proceedings of the 56th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2015.
- 20 Chen Dan, Kristoffer Arnsfelt Hansen, He Jiang, Liwei Wang, and Yuchen Zhou. On low rank approximation of binary matrices. *arXiv*, 2015. [arXiv:1511.01699](#).
- 21 A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.
- 22 Amit Deshpande and Kasturi R. Varadarajan. Sampling-based dimension reduction for subspace approximation. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing (STOC)*, pages 641–650, 2007.
- 23 Dan Feldman, Amos Fiat, Micha Sharir, and Danny Segev. Bi-criteria linear-time approximations for generalized k-mean/median/center. In *Proceedings of the 23rd Annual Symposium on Computational Geometry (SCG)*, pages 19–26, 2007.
- 24 Fedor V Fomin, Petr A Golovach, and Fahad Panolan. Parameterized low-rank binary matrix approximation. In *Proceedings of the 45th International Colloquium on Automata, Languages and Programming (ICALP)*, 2018.
- 25 Silvia Gandy, Benjamin Recht, and Isao Yamada. Tensor completion and low-rank tensor recovery via convex optimization. *Inverse Problems*, 27(2):025010, 2011.

- 26 Nicolas Gillis and Stephen A Vavasis. On the complexity of robust PCA and  $\ell_1$ -norm low-rank matrix approximation. *Mathematics of Operations Research*, 43(4):1072–1084, 2018.
- 27 Mika Göös, TS Jayram, Toniann Pitassi, and Thomas Watson. Randomized communication vs. partition number. In *Proceedings of the 44th International Colloquium on Automata, Languages and Programming (ICALP)*, 2017.
- 28 Leslie Greengard and John Strain. The fast Gauss transform. *SIAM Journal on Scientific and Statistical Computing*, 12(1):79–94, 1991.
- 29 Charles Guyon, Thierry Bouwmans, and El-Hadi Zahzah. Foreground detection based on low-rank and block-sparse matrix decomposition. *Proceedings of the 19th IEEE International Conference on Image Processing*, pages 1225–1228, 2012.
- 30 Moritz Hardt, Raghu Meka, Prasad Raghavendra, and Benjamin Weitz. Computational limits for matrix completion. In *Proceedings of the 27th Annual Conference on Computational Learning Theory (COLT)*, pages 703–725, 2014.
- 31 Daniel Hsu, Sham M Kakade, and Tong Zhang. Robust matrix decomposition with sparse corruptions. *IEEE Transactions on Information Theory*, 57(11):7221–7234, 2011.
- 32 Prateek Jain, Praneeth Netrapalli, and Sujay Sanghavi. Low-rank matrix completion using alternating minimization. In *Proceedings of the 45th Annual ACM Symposium on Theory of Computing (STOC)*, pages 665–674. ACM, 2013.
- 33 Rahul Jain and Hartmut Klauck. The partition bound for classical communication complexity and query complexity. In *Proceedings of the 25th Annual IEEE Conference on Computational Complexity (CCC)*, pages 247–258, 2010.
- 34 Rahul Jain, Troy Lee, and Nisheeth K Vishnoi. A quadratically tight partition bound for classical communication complexity and query complexity. *arXiv*, 2014. [arXiv:1401.4512](https://arxiv.org/abs/1401.4512).
- 35 Daniel M Kane, Jelani Nelson, and David P Woodruff. On the exact space complexity of sketching and streaming small norms. In *Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1161–1178, 2010.
- 36 Ravi Kannan and Santosh Vempala. Spectral algorithms. *Foundations and Trends in Theoretical Computer Science*, 2009.
- 37 Raghunandan H Keshavan, Andrea Montanari, and Sewoong Oh. Matrix completion from a few entries. *IEEE Transactions on Information Theory*, 56(6):2980–2998, 2010.
- 38 Gillat Kol, Shay Moran, Amir Shpilka, and Amir Yehudayoff. Approximate nonnegative rank is equivalent to the smooth rectangle bound. In *Proceedings of the 41st International Colloquium on Automata, Languages and Programming (ICALP)*, pages 701–712, 2014.
- 39 Eyal Kushilevitz and Noam Nisan. *Communication complexity*. Cambridge University Press, 1997.
- 40 Anastasios Kyrillidis and Volkan Cevher. Matrix ALPS: Accelerated low rank and sparse matrix reconstruction. In *2012 IEEE Statistical Signal Processing Workshop (SSP)*, pages 185–188, 2012.
- 41 Qiuwei Li, Gongguo Tang, and Arye Nehorai. Robust principal component analysis based on low-rank and block-sparse matrix decomposition. *Handbook of Robust Low-Rank and Sparse Matrix Decomposition: Applications in Image and Video Processing*, 2016.
- 42 Shuangjiang Li, Wei Wang, Hairong Qi, Bulent Ayhan, Chiman Kwan, and Steven Vance. Low-rank tensor decomposition based anomaly detection for hyperspectral imagery. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pages 4525–4529, 2015.
- 43 Ji Liu, Przemyslaw Musialski, Peter Wonka, and Jieping Ye. Tensor completion for estimating missing values in visual data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):208–220, 2013.
- 44 Antoine Liutkus and Kazuyoshi Yoshii. A diagonal plus low-rank covariance model for computationally efficient source separation. In *27th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6. IEEE, 2017.
- 45 Canyi Lu, Jiashi Feng, Yudong Chen, Wei Liu, Zhouchen Lin, and Shuicheng Yan. Tensor robust principal component analysis: Exact recovery of corrupted low-rank tensors via convex optimization. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 5249–5257, 2016.



- 46 Haibing Lu, Jaideep Vaidya, and Vijayalakshmi Atluri. Optimal Boolean matrix decomposition: Application to role engineering. In *Proceedings of the 24th IEEE International Conference on Data Engineering*, pages 297–306, 2008.
- 47 Michael W. Mahoney. Randomized algorithms for matrices and data. *Foundations and Trends in Machine Learning*, 3(2):123–224, 2011.
- 48 Antonio Valerio Miceli Barone. Low-rank passthrough neural networks. In *Proceedings of the Workshop on Deep Learning Approaches for Low-Resource NLP*, pages 77–86. Association for Computational Linguistics, 2018.
- 49 Andrew C Miller, Nicholas J Foti, and Ryan P Adams. Variational boosting: Iteratively refining posterior approximations. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 2017.
- 50 Cun Mu, Bo Huang, John Wright, and Donald Goldfarb. Square deal: Lower bounds and improved relaxations for tensor recovery. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*, pages 73–81, 2014.
- 51 Praneeth Netrapalli, U. N. Niranjan, Sujay Sanghavi, Animashree Anandkumar, and Prateek Jain. Non-convex robust PCA. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 1107–1115, 2014. URL: <http://papers.nips.cc/paper/5430-non-convex-robust-pca>.
- 52 Noam Nisan. The communication complexity of threshold gates. *Combinatorics, Paul Erdos is Eighty*, 1:301–315, 1993.
- 53 Anup Rao and Amir Yehudayoff. Communication complexity and applications (early draft), 2019. URL: <https://homes.cs.washington.edu/~anuprao/pubs/book.pdf>.
- 54 Ilya Razenshteyn, Zhao Song, and David P Woodruff. Weighted low rank approximations with provable guarantees. In *Proceedings of the 48th Annual ACM Symposium on Theory of Computing (STOC)*, 2016.
- 55 Vladimir Rokhlin. Rapid solution of integral equations of classical potential theory. *Journal of Computational Physics*, 60(2):187–207, 1985.
- 56 Donald B. Rubin and Dorothy T. Thayer. EM algorithms for ML factor analysis. *Psychometrika*, 47(1):69–76, 1982.
- 57 James Saunderson, Venkat Chandrasekaran, Pablo A Parrilo, and Alan S Willsky. Diagonal and low-rank matrix decompositions, correlation matrices, and ellipsoid fitting. *SIAM Journal on Matrix Analysis and Applications*, 33(4):1395–1416, 2012.
- 58 Tselil Schramm and Benjamin Weitz. Low-rank matrix completion with adversarial missing entries. *CoRR*, 2015.
- 59 Bao-Hong Shen, Shuiwang Ji, and Jieping Ye. Mining discrete patterns via binary matrix factorization. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 757–766, 2009.
- 60 Alexander Sherstov. Lecture notes for CS 289A Communication Complexity, 2012. URL: <http://web.cs.ucla.edu/~sherstov/teaching/2012-winter/docs/lecture05.pdf>.
- 61 Edward Snelson and Zoubin Ghahramani. Sparse Gaussian processes using pseudo-inputs. *Advances in Neural Information Processing Systems 18 (NeurIPS)*, 18:1257–1264, 2005.
- 62 Zhao Song, David P Woodruff, and Peilin Zhong. Low rank approximation with entrywise  $\ell_1$ -norm error. In *Proceedings of the 49th Annual ACM Symposium on Theory of Computing (STOC)*, pages 688–701, 2017.
- 63 Zhao Song, David P Woodruff, and Peilin Zhong. Relative error tensor low rank approximation. In *Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2772–2789. SIAM, 2019.
- 64 Charles Spearman. “General Intelligence,” objectively determined and measured. *The American Journal of Psychology*, 15(2):201–292, 1904.
- 65 Michael L. Stein. Limitations on low-rank approximations for covariance matrices of spatial data. *Spatial Statistics*, 8:1–19, 2014.

- 66 Jos MF Ten Berge and Henk AL Kiers. A numerical approach to the approximate and the exact minimum rank of a covariance matrix. *Psychometrika*, 56(2):309–315, 1991.
- 67 Jaideep Vaidya. Boolean matrix decomposition problem: theory, variations and applications to data engineering. In *Proceedings of the 28th IEEE International Conference on Data Engineering*, pages 1222–1224, 2012.
- 68 Stef van Buuren. *Flexible imputation of missing data*. Chapman and Hall/CRC, 2018. URL: <https://stefvanbuuren.name/fimd/missing-data-pattern.html>.
- 69 Shusen Wang, Chao Zhang, Hui Qian, and Zhihua Zhang. Improving the modified Nyström method using spectral shifting. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2014.
- 70 David P. Woodruff. Sketching as a tool for numerical linear algebra. *Foundations and Trends in Theoretical Computer Science*, 10(1-2):1–157, 2014.
- 71 John Wright, Arvind Ganesh, Shankar Rao, Yigang Peng, and Yi Ma. Robust principal component analysis: Exact recovery of corrupted low-rank matrices via convex optimization. In *Advances in Neural Information Processing Systems 22: 23rd Annual Conference on Neural Information Processing Systems 2009. Proceedings of a meeting held 7-10 December 2009, Vancouver, British Columbia, Canada*. Curran Associates, Inc., 2009.
- 72 Changjiang Yang, Ramani Duraiswami, Nail A Gumerov, and Larry Davis. Improved fast Gauss transform and efficient kernel density estimation. In *Proceedings of the 9th IEEE International Conference on Computer Vision*, page 464, 2003.
- 73 Xiao Zhang, Lingxiao Wang, and Quanquan Gu. A unified framework for nonconvex low-rank plus sparse matrix recovery. In *Proceedings of the 21st International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 84, 2018.
- 74 Yong Zhao, Jinyu Li, and Yifan Gong. Low-rank plus diagonal adaptation for deep neural networks. In *Proceedings of the 2016 International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 5005–5009, 2016.

# Pseudorandom Generators for Unbounded-Width Permutation Branching Programs

William M. Hoza 

University of Texas at Austin, TX, USA

<https://williamhoza.com/tcs/>

whoza@utexas.edu

Edward Pyne

Harvard University, Cambridge, MA, USA

<https://sites.google.com/view/tedpyne/>

epyne@college.harvard.edu

Salil Vadhan

Harvard University, Cambridge, MA, USA

<https://salil.seas.harvard.edu/>

salil\_vadhan@harvard.edu

---

## Abstract

We prove that the Impagliazzo-Nisan-Wigderson [9] pseudorandom generator (PRG) fools ordered (read-once) permutation branching programs of *unbounded width* with a seed length of  $\tilde{O}(\log d + \log n \cdot \log(1/\varepsilon))$ , assuming the program has only one accepting vertex in the final layer. Here,  $n$  is the length of the program,  $d$  is the degree (equivalently, the alphabet size), and  $\varepsilon$  is the error of the PRG. In contrast, we show that a randomly chosen generator requires seed length  $\Omega(n \log d)$  to fool such unbounded-width programs. Thus, this is an unusual case where an explicit construction is “better than random.”

Except when the program’s width  $w$  is very small, this is an improvement over prior work. For example, when  $w = \text{poly}(n)$  and  $d = 2$ , the best prior PRG for permutation branching programs was simply Nisan’s PRG [15], which fools general ordered branching programs with seed length  $O(\log(wn/\varepsilon) \log n)$ . We prove a seed length lower bound of  $\tilde{\Omega}(\log d + \log n \cdot \log(1/\varepsilon))$  for fooling these unbounded-width programs, showing that our seed length is near-optimal. In fact, when  $\varepsilon \leq 1/\log n$ , our seed length is within a constant factor of optimal. Our analysis of the INW generator uses the connection between the PRG and the derandomized square of Rozenman and Vadhan [20] and the recent analysis of the latter in terms of unit-circle approximation by Ahmadinejad et al. [1].

**2012 ACM Subject Classification** Theory of computation → Pseudorandomness and derandomization

**Keywords and phrases** Pseudorandom generators, permutation branching programs

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.7

**Funding** *William M. Hoza:* Supported by the NSF GRFP under grant DGE-1610403 and a Harrington Fellowship from UT Austin.

*Edward Pyne:* Supported by NSF grant CCF-1763299.

*Salil Vadhan:* Supported by NSF grant CCF-1763299 and a Simons Investigator Award.

**Acknowledgements** We thank Jack Murtagh for collaboration at the start of this research. The first author thanks Dean Doron for insightful and relevant discussions about the works by Murtagh et al. [13, 14]. We thank Shyam Narayanan, Dean Doron and David Zuckerman for valuable comments on a draft of this paper.



© William M. Hoza, Edward Pyne, and Salil Vadhan;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 7; pp. 7:1–7:20



Leibniz International Proceedings in Informatics

LIPIcs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

## 1 Introduction

Randomness, like time or space, is a computational resource. All else being equal, it is best to use as few random bits as possible. A *pseudorandom generator* (PRG) is a tool for reducing the number of random bits used by some computational process.

► **Definition 1.1.** Let  $\mathcal{F}$  be a class of functions  $B: [d]^n \rightarrow \{0, 1\}$ . An  $\epsilon$ -PRG for  $\mathcal{F}$  is a function  $G: \{0, 1\}^s \rightarrow [d]^n$  such that for every  $B \in \mathcal{F}$ ,

$$|\Pr[B(U_{[d]^n}) = 1] - \Pr[B(G(U_{\{0,1\}^s})) = 1]| \leq \epsilon,$$

where  $U_S$  is the uniform distribution over the set  $S$ . The value  $s$  is the **seed length** of the PRG.

Motivated by the goal of derandomizing small-space computation, a long line of research has studied PRGs for classes  $\mathcal{F}$  of functions computable by *branching programs*.

► **Definition 1.2.** An ordered **branching program**  $B$  of length  $n$ , width  $w$  and degree  $d$  computes a function  $B: [d]^n \rightarrow \{0, 1\}$ . At time step  $t \in [n]$ , the program maintains a state in  $[w]$ , reads the next symbol  $\sigma_t$  of the input  $\sigma \in [d]^n$  and updates its state according to a transition function  $W_t: [w] \times [d] \rightarrow [w]$ . We allow the transition function  $W_t$  to be different at each time step.

Moreover, there is an initial state  $v_{\text{start}} \in [w]$  and a single accept state  $v_{\text{end}} \in [w]$ . Let  $u$  be the final state of the branching program on input  $\sigma$ . If  $u = v_{\text{end}}$  the branching program accepts, denoted  $B(\sigma) = 1$ . For any other final state the program rejects, denoted  $B(\sigma) = 0$ .

We can represent a branching program as a graph, with  $n + 1$  layers and  $w$  vertices per layer corresponding to the states of the program at each step. For all  $t \in [n]$ , for state  $s$  in layer  $t - 1$  and  $s'$  in layer  $t$ , we add edge  $(s, s')$  with label  $\sigma_t \in [d]$  if  $W_t(s, \sigma_t) = s'$ . An ordered read-once branching program of length  $n$  and width  $w$  can compute the output of an algorithm that uses  $\log w$  bits of memory and  $n$  random bits, by taking the state at each layer as the contents of memory at that time. Unusually, we will consider branching programs where the width is unbounded (e.g., it can even be  $w = d^n$ ), albeit with the restriction of being a *permutation branching program*.

► **Definition 1.3.** A **permutation branching program** is an ordered branching program where for all  $t \in [n]$  and  $\sigma \in [d]$ ,  $W_t(\cdot, \sigma)$  is a permutation. This can be thought of as the computation being time-reversible.

Note that with this restriction the graph representation consists of  $n + 1$  layers where each layer is the union of  $d$  perfect matchings, with each matching corresponding to a distinct input symbol.

Restricted classes of branching programs, including permutation branching programs [22, 5, 10], have received attention largely because of the lack of progress on designing PRGs for general length- $n$  width- $n$  branching programs since the work of Nisan three decades ago [15]. There has also been work on permutation branching programs where the input is read in an arbitrary order [17, 3]. Our main theorem is that there is an explicit PRG fooling unbounded-width permutation branching programs with seed length that is nearly logarithmic in  $n$  and has no dependence on the width  $w$ :

► **Theorem 1.4 (Main Theorem).** For all  $n, d \in \mathbb{N}$  and  $\epsilon > 0$ , there is an explicitly computable  $\epsilon$ -PRG  $G: \{0, 1\}^s \rightarrow [d]^n$  for permutation branching programs of length  $n$ , degree  $d$ , and arbitrary width. This PRG has seed length

$$O(\log d + \log n \cdot (\log \log n + \log(1/\epsilon))).$$

In contrast, we show that a randomly chosen generator requires seed length  $\Omega(n \log d)$  to fool such unbounded-width programs. Thus, this is an unusual case where an explicit construction is “better than random.” (See Section 1.3 for more discussion.)

The PRG is an instantiation of the Impagliazzo-Nisan-Wigderson (INW) generator [9]. The proof uses the interpretation of the INW generator in terms of the derandomized square for consistently labeled graphs, introduced by Rozenman and Vadhan [20], and the analysis of the derandomized square in terms of unit-circle approximation by Ahmadinejad et al. [1].

We emphasize that our definition of permutation branching program only allows one accepting vertex. This assumption is crucial: a permutation branching program with unbounded width and an unbounded number of accepting vertices can compute any Boolean function on  $[d]^n$ , so nontrivial PRGs for that model do not exist. That being said, a program with  $a$  accepting vertices can be written as a sum of  $a$  programs with one accepting vertex each, so our PRG fools such a permutation branching program with seed length

$$O(\log d + \log n \cdot (\log \log n + \log(a/\varepsilon))).$$

## 1.1 Prior Work on the Derandomized Square

In the paper introducing the derandomized square [20], Rozenman and Vadhan showed how to use it to decide undirected connectivity in deterministic log space, giving another proof of Reingold’s famous theorem [16]. As another application, they showed how to take a (polynomially long) pseudorandom walk through a regular, aperiodic directed graph in such a way that the final vertex is distributed nearly uniformly (i.e., is close to the stationary distribution of a truly random walk), matching a result of Reingold, Trevisan, and Vadhan [18]. As mentioned previously, they observed that this pseudorandom walk is described by the INW generator, assuming the graph is “consistently labeled” (see Definition 3.4). However, their analysis does not show how to approximate short random walks (e.g., shorter than the mixing time).

In a pair of relatively recent works [13, 14], Murtagh et al. showed how to approximate (in some respects) random walks of any length  $n$ , even if  $n$  is much smaller than the graph’s mixing time. These algorithms are only for undirected graphs, but the recent work by Ahmadinejad et al. [1] handles the more general case of Eulerian digraphs (as well as getting stronger results for undirected graphs). Among other tools, all three of these papers [13, 14, 1] use the derandomized square.

Fooling branching programs amounts to approximating *bounded-length* random walks through *directed* graphs, which is why we rely on Ahmadinejad et al.’s results [1] for our theorem. One of their results is a deterministic non-black-box algorithm for estimating the acceptance probability of a given polynomial-width “regular” branching program in space  $\tilde{O}(\log n)$  to within error  $\varepsilon = 1/\text{poly}(n)$ . Our theorem solves the more challenging black-box derandomization problem, although it only works for permutation branching programs and we have a worse dependence on the error parameter  $\varepsilon$ .

## 1.2 Prior PRGs for Permutation Branching Programs

Our PRG is superior to prior generators for permutation branching programs<sup>1</sup> when the width of the branching program is not small. Previous work has focused on the constant-width case. In that regime, the best PRG for permutation branching programs is due to Steinke [22].

---

<sup>1</sup> In this discussion of prior work, we focus on the case  $d = 2$  for simplicity.

He achieves seed length  $O(w^4 \log w \log n + \log n \log(1/\varepsilon))$ , which is better than our seed length by a factor of  $\log \log n$ . For larger widths up to  $w = \text{poly}(n)$ , the best prior PRG for permutation branching programs is by Braverman et al. [2], who gave a PRG for regular branching programs with seed length

$$O(\log w \log n + \log n \cdot (\log \log n + \log(1/\varepsilon))).$$

Note that when  $w = \text{poly}(n)$  and  $\varepsilon = \Omega(1)$ , Braverman et al.'s PRG has seed length  $\Theta(\log^2 n)$ , just like Nisan's PRG [15], whereas our PRG has seed length  $\tilde{O}(\log n)$ . The case  $w = \text{poly}(n)$  is arguably the most important case, because polynomial-width ordered branching programs correspond to uniform randomized algorithms that always halt. Recall that low-error PRGs for polynomial-width regular branching programs suffice for derandomizing all of **RL** [18].

When the width is even larger than  $\text{poly}(n)$ , the best prior PRG is by De [5]. De's work is focused on the constant-width case, but he also gave a generator with seed length  $O(\log(n/\varepsilon) \log n)$  independent of  $w$ .

### 1.3 Failure of the Probabilistic Method

There is something counterintuitive about the superpolynomial-width regime. Recall that for typical models of computation, including polynomial-width degree-2 branching programs, it is straightforward to show that there exists a nonexplicit PRG with seed length  $O(\log(n/\varepsilon))$ , because a random function is a good PRG. Furthermore, it is typically fairly trivial to prove a matching  $\Omega(\log(n/\varepsilon))$  lower bound. The main challenge, in most cases, is to devise an explicit construction matching the parameters of the probabilistic existence proof.

However, the standard nonexplicit existence argument is not applicable to unbounded-width permutation branching programs, because they can compute doubly-exponentially many distinct functions; in particular, we show (Lemma 5.1) that they can compute every Boolean function  $B(x, y)$  that tests whether  $\pi(x) = y$  for a permutation  $\pi: [d]^{n/2} \rightarrow [d]^{n/2}$ . And indeed, as mentioned previously, for seed length less than  $(n \log d)/4$ , we show (Theorem 5.2) that a random function is *not* a good PRG for this model. The reason is that when a generator is chosen at random, with high probability, there is some permutation  $\pi$  such that every output  $(x, y)$  of the generator satisfies  $\pi(x) = y$ .

Since the probabilistic method fails here, it might be surprising that there even *exists* a PRG with near-logarithmic seed length, let alone our explicit construction. Intuitively, the INW generator manages to outperform the probabilistic method because the second half of the INW generator's output is *information-theoretically* unpredictable given the first half, and vice versa.

We remark that another family of unbounded-width ordered branching programs has been studied previously: "monotone" branching programs. From Meka and Zuckerman's work [12], it follows that a random function *is* a good PRG for unbounded-width monotone branching programs. In that respect, the model we study is more unusual.

### 1.4 The Optimal Seed Length

These considerations raise the question of what the optimal seed length is for our model. We prove (Theorem 6.1) that any PRG for unbounded-width permutation branching programs must have seed length at least  $\Omega(\log d + \log n \cdot \log(1/\varepsilon))$ , provided  $\varepsilon$  is not extremely small.<sup>2</sup> Thus, our explicit PRG's seed length is near-optimal. In fact, although the lower bound gets

<sup>2</sup> E.g., any  $\varepsilon \geq \exp(-(n \log d)^{0.99})$  is large enough.



slightly weaker when  $\varepsilon$  is extremely small, we give a matching refinement of our *upper* bound in that regime (see Corollary 4.9), providing an explicit PRG with asymptotically optimal seed length whenever  $\varepsilon \leq 1/\log n$ .

To the best of our knowledge, this is the first known case where, e.g., some seed length  $s$  is sufficient for a constant-error PRG, but seed length  $O(s + \log n)$  is not sufficient to achieve error  $1/n$ . In the context of fooling shallow circuits, similar lower bounds were proven previously for restricted classes of PRGs such as  $k$ -wise independent distributions [11] or small-bias distributions [6], but our lower bound holds for any PRG whatsoever. Our lower bound uses basic tools from matching theory and information theory.

On the other hand, we show (Theorem 7.1) that a random function is at least a good *hitting set* generator (HSG); the optimal seed length for nonexplicit HSGs for unbounded-width permutation branching programs is  $\Theta(\log(nd/\varepsilon))$ . This is the first case we are aware of where there is a large gap between the best possible PRGs and the best possible HSGs.

## 1.5 Organization

In Section 2, we introduce measures of spectral approximation for matrices and basic linear algebra facts. In Section 3, we introduce the derandomized square, and recall two theorems relating the square to unit-circle approximation, then prove repeated derandomized squaring provides a suitable quality approximation. In Section 4, we use the bounds on repeated derandomized squares to analyze the INW generator. In Section 5, we prove that a random function with seed length less than  $(n \log d)/4$  does not fool unbounded-width permutation branching programs. In Section 6, we prove our lower bound on the seed length of any PRG for these programs. Finally, in Section 7, we identify the optimal seed length for nonexplicit HSGs for these programs.

## 2 Spectral Approximation Preliminaries

We first introduce basic notation and recall two measures of closeness of approximation for matrices, complex spectral approximation and unit-circle approximation.

- For a complex number  $z \in \mathbb{C}$  we write  $z^*$  to denote the complex conjugate of  $z$  and  $|z|$  to denote the magnitude of  $z$ .
- For a matrix  $A \in \mathbb{C}^{N \times N}$  we write  $A^*$  to denote its conjugate transpose and write  $U_A = (A + A^*)/2$  to denote its **symmetrization**.
- We say a Hermitian matrix  $A$  is **positive semidefinite** (PSD) or write  $A \succeq 0$  if  $x^*Ax \geq 0$  for all  $x \in \mathbb{C}^N$ . For two Hermitian matrices  $A, B$ , we use  $A \succeq B$  to denote  $A - B \succeq 0$  and define  $\preceq$  analogously.

► **Definition 2.1** (Complex Spectral Approximation [1]). *For  $A, B \in \mathbb{C}^{N \times N}$  and  $\varepsilon > 0$ , we say  $A$  is a **complex  $\varepsilon$ -approximation** of  $B$ , denoted  $A \approx_\varepsilon B$ , if*

$$\forall x, y \in \mathbb{C}^N, \quad |x^*(B - A)y| \leq \frac{\varepsilon}{2}(|x|^2 + |y|^2 - x^*U_Bx - y^*U_By).$$

*For two  $N$ -vertex digraphs  $\tilde{\mathbf{G}}, \mathbf{G}$  with random walk matrices  $A, B$ , write  $\tilde{\mathbf{G}} \approx_\varepsilon \mathbf{G}$  if  $A \approx_\varepsilon B$ .*

We now recall the stronger notion that we will use for analyzing the generator.

► **Definition 2.2** (Unit-Circle Approximation [1]). *For  $A, B \in \mathbb{C}^{N \times N}$  and  $\varepsilon > 0$ , we say  $A$  is a **unit-circle  $\varepsilon$ -approximation** of  $B$ , denoted  $A \overset{\circ}{\approx}_\varepsilon B$ , if*

$$\forall x, y \in \mathbb{C}^N, \quad |x^*(B - A)y| \leq \frac{\varepsilon}{2}(|x|^2 + |y|^2 - |x^*Bx + y^*By|).$$

*For two  $N$ -vertex digraphs  $\tilde{\mathbf{G}}, \mathbf{G}$  with random walk matrices  $A, B$ , write  $\tilde{\mathbf{G}} \overset{\circ}{\approx}_\varepsilon \mathbf{G}$  if  $A \overset{\circ}{\approx}_\varepsilon B$ .*

Including the magnitude operation in the right hand side forces the approximation to be exact for all eigenspaces with eigenvalues of complex magnitude 1, and this property is essential for the preservation of approximation under high powers. The unit-circle approximation is developed in [1]. We rely on a convenient equivalence between unit-circle approximation and complex approximation:

► **Lemma 2.3** ([1] Lemma 3.8). *Let  $A, B \in \mathbb{C}^{N \times N}$  and  $\varepsilon > 0$ . Then  $A \overset{\circ}{\approx}_{\varepsilon} B$  if and only if for all  $z \in \mathbb{C}$  with  $|z| = 1$ ,  $zA \approx_{\varepsilon} zB$ .*

We will also use this basic result about complex approximation. Note that Cohen et al. [4] prove the analogous statement where complex numbers are replaced with reals.

► **Lemma 2.4.** *Let  $A, B \in \mathbb{C}^{N \times N}$  where  $A \approx_{\varepsilon} B$ . Then  $(1 - \varepsilon)U_{I-B} \preceq U_{I-A} \preceq (1 + \varepsilon)U_{I-B}$ .*

**Proof.** Let arbitrary  $x \in \mathbb{C}^N$ . Bounding the gap between the symmetrizations via the definition of complex approximation gives

$$\begin{aligned} |x^*U_{I-B}x - x^*U_{I-A}x| &= \left| \frac{1}{2}(x^*(B + B^*)x - x^*(A + A^*)x) \right| \\ &\leq \left| \frac{1}{2}x^*(B - A)x \right| + \left| \frac{1}{2}x^*(B^* - A^*)x \right| \\ &\leq \frac{1}{2}\varepsilon(\|x\|^2 - x^*U_Bx) + \frac{1}{2}\varepsilon(\|x\|^2 - x^*U_Bx) \\ &= \varepsilon \cdot x^*U_{I-B}x. \end{aligned}$$

This directly implies  $x^*U_{I-A}x - (1 - \varepsilon)x^*U_{I-B}x \geq 0$  and  $(1 + \varepsilon)x^*U_{I-B}x - x^*U_{I-A}x \geq 0$ . Since  $x$  was arbitrary we are done. ◀

We now state an approximate triangle inequality for unit-circle approximation, which will be a tool for bounding the error of the generator. Previously, Cohen et al. [4] proved a similar lemma regarding the real analogue of complex approximation.

► **Lemma 2.5** (Quasi-Triangle Inequality). *If  $C \overset{\circ}{\approx}_{\varepsilon_2} B \overset{\circ}{\approx}_{\varepsilon_1} A$  then  $C \overset{\circ}{\approx}_{\varepsilon_1 + \varepsilon_2 + \varepsilon_1\varepsilon_2} A$ .*

**Proof.** Let  $z \in \mathbb{C}$  satisfy  $|z| = 1$ , and let  $x, y \in \mathbb{C}^N$  be arbitrary. Since  $B \overset{\circ}{\approx}_{\varepsilon_1} A$ , by Lemma 2.3,  $zB \approx_{\varepsilon_1} zA$ , so

$$|x^*(A - B)y| \leq \frac{\varepsilon_1}{2}(x^*U_{I-zA}x + y^*U_{I-zA}y).$$

Similarly, since  $C \overset{\circ}{\approx}_{\varepsilon_2} B$ ,

$$\begin{aligned} |x^*(B - C)y| &\leq \frac{\varepsilon_2}{2}(x^*U_{I-zB}x + y^*U_{I-zB}y) \\ &\leq \frac{\varepsilon_2}{2} \cdot (1 + \varepsilon_1) \cdot (x^*U_{I-zA}x + y^*U_{I-zA}y) \end{aligned}$$

where the second inequality follows from Lemma 2.4. Therefore,

$$\begin{aligned} |x^*(A - C)y| &\leq |x^*(A - B)y| + |x^*(B - C)y| \\ &\leq \left( \frac{\varepsilon_1}{2} + \frac{\varepsilon_2}{2} + \frac{\varepsilon_1\varepsilon_2}{2} \right) \cdot (x^*U_{I-zA}x + y^*U_{I-zA}y). \end{aligned}$$

Since  $x$  and  $y$  were arbitrary, this shows that  $zC \approx_{\varepsilon_1 + \varepsilon_2 + \varepsilon_1\varepsilon_2} zA$ . Since  $z$  was arbitrary, we are done by Lemma 2.3. ◀



► **Corollary 2.6** (Iterated Quasi-Triangle Inequality). *Suppose  $\delta \leq 1$  and*

$$A_0 \overset{\circ}{\approx}_{\delta} A_1 \overset{\circ}{\approx}_{\delta} \dots \overset{\circ}{\approx}_{\delta} A_{\ell}.$$

*Then  $A_0 \overset{\circ}{\approx}_{\varepsilon} A_{\ell}$  with  $\varepsilon = \ell\delta/(1-\delta)^2$ .*

**Proof.** Applying Lemma 2.5 inductively, we get a bound of

$$\sum_{i=0}^{\ell} (\ell - i) \cdot \delta^{i+1} = \frac{\ell\delta - \delta^2 \cdot (\ell + 1 - \delta^{\ell})}{(1 - \delta)^2} \leq \frac{\ell\delta}{(1 - \delta)^2}. \quad \blacktriangleleft$$

Finally, we give a basic result used to relate unit-circle to entrywise approximation for the final generator analysis.

► **Proposition 2.7.** *Given  $A, B \in \mathbb{C}^{N \times N}$  so that  $A \overset{\circ}{\approx}_{\varepsilon} B$ , for all indices  $u, v \in [N]$ ,  $|A_{u,v} - B_{u,v}| \leq \varepsilon$ .*

**Proof.** Let  $e_u, e_v$  be the standard basis vectors with ones in coordinates  $u, v$  respectively and apply Definition 2.2:

$$|A_{u,v} - B_{u,v}| = |e_u^*(A - B)e_v| \leq \frac{\varepsilon}{2} (||e_u||^2 + ||e_v||^2 - |e_u^* B e_u + e_v^* B e_v|) \leq \varepsilon. \quad \blacktriangleleft$$

### 3 Repeated Derandomized Squaring

#### 3.1 Graph Labelings

Branching programs are closely related to graphs with *one-way labelings*.

► **Definition 3.1** (One-Way Labeling [20]). *A **one-way labeling** of a  $d$ -regular directed multigraph  $\mathbf{G}$  assigns a label in  $[d]$  to each edge  $(u, v)$  such that for every vertex  $u$ , the labels of the outgoing edges of  $u$  are distinct. If  $\mathbf{G}$  has a one-way labeling, let  $\mathbf{G}[u, i]$  denote the vertex  $v$  such that  $(u, v)$  is labeled  $i$ .*

One-way labelings are compatible with the operation of *powering* a graph. One step on  $\mathbf{G}^n$  corresponds to  $n$  steps in  $\mathbf{G}$ . The formal definition follows.

► **Definition 3.2** (Graph Powering). *Let  $\mathbf{G}$  be a  $d$ -regular directed multigraph with a one-way labeling. For  $n \geq 1$ , we recursively define  $\mathbf{G}^n$  to be a  $(d^n)$ -regular directed multigraph on the same vertex set with a one-way labeling given by*

$$\mathbf{G}^1 = \mathbf{G}$$

$$\mathbf{G}^{n+1}[v, (e_1, e_2)] = \mathbf{G}^n[\mathbf{G}[v, e_1], e_2],$$

*identifying  $[d^{n+1}] = [d] \times [d^n]$ .*

Derandomized squaring is a way of “approximating” the powers of a graph. The derandomized squaring operation is defined in terms of graphs with additional structure, namely, a *two-way labeling*.

► **Definition 3.3** (Two-Way Labeling [20]). *A **two-way labeling** of a  $d$ -regular directed multigraph  $\mathbf{G}$  assigns two labels in  $[d]$  to each edge  $(u, v)$ : one as an edge incident to  $u$  (the “outgoing label”) and one as an edge incoming to  $v$  (the “incoming label”). We require that for every vertex  $v$ , the outgoing labels of the outgoing edges of  $v$  are distinct, and the incoming labels of the incoming edges of  $v$  are distinct. If  $\mathbf{G}$  is an  $N$ -vertex graph with a two-way labeling, we define the rotation map [19, 20]  $\text{Rot}_{\mathbf{G}}: [N] \times [d] \rightarrow [N] \times [d]$  by letting  $\text{Rot}_{\mathbf{G}}(u, i) = (v, j)$  if there is an edge  $(u, v)$  with outgoing label  $i$  and incoming label  $j$ .*

Naturally, if  $\mathbf{G}$  has a two-way labeling, we think of  $\mathbf{G}$  as also having a one-way labeling given by the outgoing labels:  $\text{Rot}_{\mathbf{G}}(u, i) = (v, j) \implies \mathbf{G}[u, i] = v$ . Conversely, there is a natural way to extend any *consistent* one-way labeling (defined next) to a two-way labeling.

► **Definition 3.4** (Consistent One-Way Labeling [8]). *A **consistent one-way labeling** of a graph  $\mathbf{G}$  is a one-way labeling such that for every vertex  $v$ , the labels of the incoming edges of  $v$  are distinct. Equivalently,  $\mathbf{G}[u, i] = \mathbf{G}[v, i] \implies u = v$ . If  $\mathbf{G}$  has a consistent one-way labeling, then we can extend  $\mathbf{G}$  to a graph  $\overline{\mathbf{G}}$  that has a two-way labeling given by*

$$\text{Rot}_{\overline{\mathbf{G}}}(u, i) = (\mathbf{G}[u, i], i).$$

### 3.2 Derandomized Squaring

Now we are ready to define the derandomized square operation, introduced by Rozenman and Vadhan [20]. Let  $\mathbf{G} = (V, E)$  be a regular directed multigraph. In the true square  $\mathbf{G}^2$ , for each vertex  $v \in V$ , there is a complete bipartite graph from in-neighbors of  $v$  to outneighbors of  $v$ , equivalent to all two-step walks through  $v$ . A derandomized square picks out a pseudorandom subset of such walks by correlating the two steps via edges on an expander graph  $\mathbf{H}$ .

► **Definition 3.5** (Derandomized Square [20]). *Let  $\mathbf{G}$  be a directed  $d$ -regular multigraph on  $N$  vertices with a two-way labeling. Let  $\mathbf{H}$  be a directed  $c$ -regular multigraph on  $d$  vertices with a one-way labeling. We define the **derandomized square**  $\mathbf{G} \circledast \mathbf{H}$  to be a  $(cd)$ -regular directed multigraph on  $N$  vertices with a one-way labeling given by*

$$(\mathbf{G} \circledast \mathbf{H})[v, (i, j)] = \mathbf{G}[v', \mathbf{H}[i', j]],$$

where  $(v', i') = \text{Rot}_{\mathbf{G}}(v, i)$ .

Note that Definition 3.5 requires  $\mathbf{G}$  to have a two-way labeling, but the derandomized square  $\mathbf{G} \circledast \mathbf{H}$  itself only has a one-way labeling. If we wish to apply the derandomized squaring operation a second time to approximate  $\mathbf{G}^4$ , we must first assign incoming labels to the edges in  $\mathbf{G} \circledast \mathbf{H}$ . When they introduced the derandomized square operation, Rozenman and Vadhan studied two distinct approaches for assigning incoming edge labels [20]. The first approach is to assume that we start with a graph  $\mathbf{G}$  with a consistent one-way labeling. In this case,  $\overline{\mathbf{G}} \circledast \mathbf{H}$  has a consistent one-way labeling as well (see Lemma 4.2). This approach is closely connected to the INW generator [9], as we will discuss in Section 4. The second approach is to assume that  $\mathbf{H}$  has a two-way labeling. In this case, one can assign incoming edge labels to  $\mathbf{G} \circledast \mathbf{H}$  by setting  $\text{Rot}_{\mathbf{G} \circledast \mathbf{H}}(v_0, (i_0, j_0)) = (v_2, (i_3, j_1))$ , where

$$(v_1, i_1) = \text{Rot}_{\mathbf{G}}(v_0, i_0), \quad (i_2, j_1) = \text{Rot}_{\mathbf{H}}(i_1, j_0), \quad (v_2, i_3) = \text{Rot}_{\mathbf{G}}(v_1, i_2).$$

This is the approach taken in, e.g., the recent work of Ahmadi et al. [1]. Note that if  $\mathbf{G}$  has a consistent one-way labeling and  $\mathbf{H}$  has a two-way labeling, the two approaches for assigning incoming edge labels to  $\overline{\mathbf{G}} \circledast \mathbf{H}$  do not coincide.

Like previous work, we will use auxiliary graphs  $\mathbf{H}$  that are good *spectral expanders*, meaning that  $\lambda(\mathbf{H})$  (defined next) is small. For the purposes of this paper, an *undirected* graph is a symmetric directed graph, i.e., a directed graph such that for every edge  $(u, v)$ , the reverse edge  $(v, u)$  is also present.

► **Definition 3.6.** *Let  $\mathbf{H}$  be an undirected regular multigraph on  $N$  vertices with random walk matrix  $M$ . We define  $\lambda(\mathbf{H}) = \max_{x \in \mathbb{R}^N: \langle \mathbf{1}, x \rangle = 0} \|Mx\|_2 / \|x\|_2$ , where  $\mathbf{1}$  is the all-ones vector. This is equal to the second largest eigenvalue in absolute value of  $M$ .*

Ahmadinejad et al. showed that the derandomized square is a unit-circle approximation of the true square [1]. Since the conclusion of this theorem is only a statement about the random walk matrix of  $\mathbf{G} \circledast \mathbf{H}$ , the theorem is oblivious to any edge labels in  $\mathbf{G} \circledast \mathbf{H}$ .

► **Theorem 3.7** ([1] Theorem 5.9). *Let  $\mathbf{G}$  be a  $d$ -regular directed multigraph with a two-way labeling, and let  $\mathbf{H}$  be a  $c$ -regular undirected multigraph on  $d$  vertices with a one-way labeling. If  $\lambda(\mathbf{H}) \leq \varepsilon$ , then  $\mathbf{G} \circledast \mathbf{H} \overset{\circ}{\approx}_{2\varepsilon} \mathbf{G}^2$ .*

We now use the spectral approximation measures of Section 2 to bound the error introduced by repeated derandomized squares. In the theorem below, although  $\mathbf{G}_i$  has a two-way labeling, when we write  $\mathbf{G}_i = \mathbf{G}_{i-1} \circledast \mathbf{H}_i$ , we merely mean equality of one-way labelings. Thus, our bound applies *regardless* of how the incoming edge labels of  $\mathbf{G}_{i-1} \circledast \mathbf{H}_i$  are assigned, as long as they form a valid two-way labeling.

► **Theorem 3.8** (Repeated Derandomized Squaring). *Let  $\mathbf{G}_0, \mathbf{G}_1, \dots, \mathbf{G}_\ell$  be directed multigraphs on  $N$  vertices with two-way labelings, where  $\mathbf{G}_i$  is  $(d \cdot c^i)$ -regular. Let  $\varepsilon \in (0, 0.12)$ , and let  $\mathbf{H}_1, \dots, \mathbf{H}_\ell$  be undirected  $c$ -regular multigraphs with one-way labelings, where  $\mathbf{H}_i$  is on  $d \cdot c^{i-1}$  vertices and  $\lambda(\mathbf{H}_i) \leq \varepsilon$ . Assume that for every  $i \in [\ell]$ , we have  $\mathbf{G}_i = \mathbf{G}_{i-1} \circledast \mathbf{H}_i$ . Then  $\mathbf{G}_\ell \overset{\circ}{\approx}_{8\ell\varepsilon} \mathbf{G}_0^{2^\ell}$ .*

The proof of Theorem 3.8 relies on a result by Ahmadinejad et al. [1] saying that unit-circle approximations are preserved under arbitrary *true* powers.

► **Lemma 3.9** ([1] Corollary 4.9). *Let  $\tilde{\mathbf{G}}, \mathbf{G}$  be directed multigraphs. If  $\tilde{\mathbf{G}} \overset{\circ}{\approx}_\varepsilon \mathbf{G}$  then for all  $k \in \mathbb{N}$  we have  $\tilde{\mathbf{G}}^k \overset{\circ}{\approx}_{\varepsilon/(1-\frac{3}{2}\varepsilon)} \mathbf{G}^k$ .*

**Proof of Theorem 3.8.** By Theorem 3.7, for all  $j$ ,  $\mathbf{G}_{i+1} \overset{\circ}{\approx}_{2\varepsilon} \mathbf{G}_i^2$ . We then use Lemma 3.9 which states that we can take arbitrary powers and preserve unit-circle approximation. For arbitrary  $i \in [\ell]$ ,  $k_i \in \mathbb{N}$  we have  $\mathbf{G}_{i+1}^{k_i} \overset{\circ}{\approx}_{\frac{2\varepsilon}{1-3\varepsilon}} \mathbf{G}_i^{2k_i}$ . Then by choosing  $k_i = 2^{\ell-i}$  we obtain a chain

$$\mathbf{G}_\ell \overset{\circ}{\approx}_{\frac{2\varepsilon}{1-3\varepsilon}} \mathbf{G}_{\ell-1}^2 \overset{\circ}{\approx}_{\frac{2\varepsilon}{1-3\varepsilon}} \mathbf{G}_{\ell-2}^4 \overset{\circ}{\approx}_{\frac{2\varepsilon}{1-3\varepsilon}} \dots \overset{\circ}{\approx}_{\frac{2\varepsilon}{1-3\varepsilon}} \mathbf{G}_0^{2^\ell},$$

relating the final derandomized square to the true power via a sequence of unit-circle approximations. Applying Corollary 2.6 gives the bound  $\mathbf{G}_\ell \overset{\circ}{\approx}_C \mathbf{G}_0^{2^\ell}$  where  $C = 2\varepsilon\ell \cdot \frac{1-3\varepsilon}{(1-5\varepsilon)^2} \leq 8\varepsilon\ell$ . ◀

## 4 The Pseudorandom Generator

In this section, we present the PRG of Theorem 1.4. We first state the definition of the Impagliazzo-Nisan-Wigderson (INW) generator and relate it to the repeated derandomized square. For the remainder of the section, fix a sequence of  $c$ -regular undirected multigraphs  $\mathbf{H}_1, \mathbf{H}_2, \dots$  where  $\mathbf{H}_i$  has  $d \cdot c^{i-1}$  vertices and has a one way labeling. We define a sequence of generators  $\text{INW}_0, \text{INW}_1, \dots$  such that  $\text{INW}_i : [d] \times [c]^i \rightarrow [d]^{2^i}$ .

► **Definition 4.1** (INW Generator [9]). *Define  $\text{INW}_0(\sigma) = \sigma$  for  $\sigma \in [d]$  as the trivial PRG that outputs its input and  $\text{INW}_{i+1}(v, e) = (\text{INW}_i(v), \text{INW}_i(\mathbf{H}_{i+1}[v, e]))$ .*

This is the recursive definition of the INW generator [9]. However, in the context of graphs with consistent one-way labelings there exists an equivalent characterization in terms of the derandomized square [20], which we will use for our analysis. The following two lemmas follow from the reasoning in Rozenman and Vadhan's work [20, Theorem 5.8]. We repeat the proofs here for completeness.

► **Lemma 4.2.** *Let  $\mathbf{G}$  be a  $d$ -regular multigraph and  $\mathbf{H}$  a  $c$ -regular undirected multigraph on  $d$  vertices. If  $\mathbf{G}$  has a consistent one-way labeling, then  $\overline{\mathbf{G}} \circledast \mathbf{H}$  has a consistent one-way labeling.*

**Proof.** Let  $\tilde{\mathbf{G}} = \overline{\mathbf{G}} \circledast \mathbf{H}$ . By the definitions of  $\overline{\mathbf{G}}$  and  $\circledast$ , we have

$$\tilde{\mathbf{G}}[v, (i, j)] = \mathbf{G}[\mathbf{G}[v, i], \mathbf{H}[i, j]].$$

To prove that  $\tilde{\mathbf{G}}$  has a consistent one-way labeling, fix  $(i, j)$ , and suppose  $\tilde{\mathbf{G}}[u, (i, j)] = \tilde{\mathbf{G}}[v, (i, j)]$ . We must show that  $u = v$ . Indeed,  $\mathbf{G}[\mathbf{G}[u, i], \mathbf{H}[i, j]] = \mathbf{G}[\mathbf{G}[v, i], \mathbf{H}[i, j]]$ . Since  $\mathbf{G}$  has a consistent one-way labeling, this implies that  $\mathbf{G}[u, i] = \mathbf{G}[v, i]$ . Again using the fact that  $\mathbf{G}$  has a consistent one-way labeling, this implies that  $u = v$  as desired. ◀

► **Lemma 4.3.** *Let  $\mathbf{G}_0$  be a  $d$ -regular multigraph on any number of vertices with a consistent one-way labeling. For  $i \geq 0$ , inductively define  $\mathbf{G}_{i+1} = \overline{\mathbf{G}_i} \circledast \mathbf{H}_{i+1}$ . Then for all  $v$  and  $e$ ,  $\mathbf{G}_i[v, e] = \mathbf{G}_0^{2^i}[v, \text{INW}_i(e)]$ .*

**Proof.** First, note that inductively,  $\mathbf{G}_i$  has a consistent labeling by Lemma 4.2, so  $\mathbf{G}_{i+1}$  is well-defined. Now we show by induction on  $i$  that  $\mathbf{G}_i[v, e] = \mathbf{G}_0^{2^i}[v, \text{INW}_i(e)]$ . The case of  $\mathbf{G}_0$  is immediate. Assume the inductive hypothesis holds for  $i$ . Fix an arbitrary vertex  $v$  and end label  $e = (e_1, e_2) \in [d \cdot c^i] \times [c]$ . We have

$$\begin{aligned} \mathbf{G}_{i+1}[v, e] &= \mathbf{G}_i[\mathbf{G}_i[v, e_1], \mathbf{H}_{i+1}[e_1, e_2]] && \text{(Definitions)} \\ &= \mathbf{G}_0^{2^i}[\mathbf{G}_0^{2^i}[v, \text{INW}_i(e_1)], \text{INW}_i(\mathbf{H}_{i+1}[e_1, e_2])] && \text{(Induction hypothesis)} \\ &= \mathbf{G}_0^{2^{i+1}}[v, (\text{INW}_i(e_1), \text{INW}_i(\mathbf{H}_{i+1}[e_1, e_2]))] \\ &= \mathbf{G}_0^{2^{i+1}}[v, \text{INW}_{i+1}(e)]. \end{aligned} \quad \blacktriangleleft$$

Let  $\ell = \lceil \log(n) \rceil$ , and define  $G: [d] \times [c]^\ell \rightarrow [d]^n$  by letting  $G(x)$  be the  $n$ -symbol prefix of  $\text{INW}_\ell(x)$ . This will be the generator that proves Theorem 1.4.

## 4.1 Approximation Guarantee

To bridge the gap between regular graphs and branching programs, we now define the *execution graph* of a branching program, which is just like the standard graph representation of the program, but the length is padded to a power of two and edges are added to wrap around from the end to the beginning.

► **Definition 4.4** (Branching Program Execution Graph). *Let  $B$  be a permutation branching program of width  $w$ , degree  $d$  and length  $n$ , and let  $m$  be the smallest power of 2 greater than  $n$ . Define the **execution graph** of  $B$  to be a directed  $d$ -regular multigraph  $\mathbf{G}$  on the vertex set  $\{0, \dots, m\} \times [w]$  with a one-way labeling given by*

$$\mathbf{G}[(t, u), \sigma] = \begin{cases} (t+1, W_t(u, \sigma)) & t \in \{0, \dots, n-1\} \\ (t+1, u) & t \in \{n, \dots, m-1\} \\ (0, u) & t = m, \end{cases}$$

where  $W_t$  is the transition function of  $B$  at layer  $i$  as in Definition 1.3.

► **Remark 4.5.** Since  $B$  is a permutation branching program, the execution graph  $\mathbf{G}$  has a consistent one-way labeling. This is not true for general regular branching programs and is why our method does not generalize.

▷ **Claim 4.6.** Let  $\varepsilon > 0$ . If every  $\mathbf{H}_i$  satisfies  $\lambda(\mathbf{H}_i) \leq \varepsilon/(8\ell)$ , then  $G$  is an  $\varepsilon$ -PRG for permutation branching programs of degree  $d$ , length  $n$ , and arbitrary width.

*Proof.* Let  $B$  be an arbitrary permutation branching program of degree  $d$  and length  $n$ . Let  $\mathbf{G}_0$  be the execution graph of  $B$ . Let  $\mathbf{G}_1, \mathbf{G}_2, \dots$  be the graphs in Lemma 4.3. By Theorem 3.8 we have  $\mathbf{G}_\ell \overset{\circ}{\approx}_\varepsilon \mathbf{G}_0^m$ .

Let  $u = (0, v_{\text{start}})$  be the start vertex in the execution graph, and let  $v = (m, v_{\text{end}})$  be the accept vertex. By the definition of  $\mathbf{G}_0$ , for all  $\sigma \in [d]^m$ , we have  $\mathbf{G}_0^m[u, \sigma] = (m, a)$ , where  $a$  is the final state of  $B$  when it reads  $\sigma_{1\dots n}$ . Therefore,  $\mathbf{G}_0^m[u, \sigma] = v \iff B(\sigma_{1\dots n}) = 1$ .

Let  $M$  be the random walk matrix of  $\mathbf{G}_0$  and  $\widetilde{M}$  the random walk matrix of  $\mathbf{G}_\ell$ . Then

$$\begin{aligned} & \left| \Pr_{x \leftarrow U_{[d] \times [c]^\ell}} [B(G(x)) = 1] - \Pr_{\sigma \leftarrow U_{[d]^m}} [B(\sigma) = 1] \right| \\ &= \left| \Pr_{x \leftarrow U_{[d] \times [c]^\ell}} [\mathbf{G}_0^m[u, \text{INW}_\ell(x)] = v] - \Pr_{e \leftarrow U_{[d]^m}} [\mathbf{G}_0^m[u, e] = v] \right| \\ &= \left| \Pr_{e \leftarrow U_{[d] \times [c]^\ell}} [\mathbf{G}_\ell[u, e] = v] - \Pr_{e \leftarrow U_{[d]^m}} [\mathbf{G}_0^m[u, e] = v] \right| \\ &= \left| \widetilde{M}_{v,u} - M_{v,u}^m \right|, \end{aligned}$$

which is at most  $\varepsilon$  by Proposition 2.7.  $\triangleleft$

To complete the proof of Theorem 1.4 we recall a result giving the existence of explicit expanders of all sizes.

▷ **Lemma 4.7** ([14] Theorem 3.3, Definition 2.13). *For all  $n > 1$  and  $\lambda > 0$ , there is a  $c = \text{poly}(1/\lambda)$  and a  $c$ -regular undirected multigraph  $\mathbf{H}$  on  $n$  vertices with a one-way labeling such that  $\lambda(\mathbf{H}) \leq \lambda$ , and given  $\lambda$ ,  $v$ , and  $e$ , the vertex  $\mathbf{H}[v, e]$  can be computed in space  $O(\log(nc))$ .*

**Proof of Theorem 1.4.** Let  $\mathbf{H}_i$  be the expander given by Lemma 4.7 with  $\lambda = \varepsilon/(8\ell)$  and  $n = d \cdot c^{i-1}$ . This sequence  $\mathbf{H}_1, \mathbf{H}_2, \dots$  satisfies the requirements of Claim 4.6, so  $G$  constructed with this sequence is an  $\varepsilon$ -PRG. It remains to show the seed length and that the generator is explicit.

By construction the  $\ell$ th INW generator  $\text{INW}_\ell$  has domain  $[d] \times [c]^\ell$ . By definition  $\ell \leq \log(n) + 1$  and by Lemma 4.7 the degree of  $\mathbf{H}_i$  for all  $i$  is  $c = \text{poly}(\log(n)/\varepsilon)$ , which gives a seed length of  $s = O(\log d + \log n \cdot (\log \log n + \log(1/\varepsilon)))$ .

Finally,  $G$  is explicit, in that the output of the generator can be computed in working space  $O(s)$ . This follows directly from Definition 4.1 and the explicitness of the expanders.  $\blacktriangleleft$

## 4.2 Improved Seed Length for Tiny Error

So far, we have designed a PRG with seed length

$$O(\log d + \log n \cdot (\log \log n + \log(1/\varepsilon))). \quad (1)$$

In this section, we will present a simple reduction that yields an improved seed length when  $\varepsilon$  is extremely small.

▷ **Lemma 4.8.** *Suppose  $G: \{0, 1\}^s \rightarrow [d^m]^n$  is an  $\varepsilon$ -PRG for length- $n$  degree- $(d^m)$  permutation branching programs. Identify  $[d^m] = [d]^m$ , and think of  $G$  as a function  $G: \{0, 1\}^s \rightarrow [d]^{mn}$ . Then  $G$  is an  $\varepsilon$ -PRG for length- $(mn)$  degree- $d$  permutation branching programs.*

**Proof.** Let  $B$  be a length- $(mn)$  degree- $d$  permutation branching program. Define a length- $n$  degree- $(d^m)$  branching program  $B'$  where one step of  $B'$  simulates  $m$  steps of  $B$ . Then  $B'$  is a permutation branching program, and  $B'$  computes the same function as  $B$ , so fooling  $B'$  implies fooling  $B$ . ◀

► **Corollary 4.9.** *For all  $n, d \in \mathbb{N}$  and  $\varepsilon > d^{-n/2}$ , there is an explicitly computable  $\varepsilon$ -PRG  $G: \{0, 1\}^s \rightarrow [d]^n$  for permutation branching programs of length  $n$ , degree  $d$ , and arbitrary width. This PRG has seed length*

$$O\left(\log d + \log\left(\frac{n \log d}{\log(1/\varepsilon)}\right) \cdot (\log \log n + \log(1/\varepsilon))\right).$$

**Proof.** If  $\log(1/\varepsilon) < \log d$ , then the seed length of Equation 1 is already sufficient. Assume, therefore, that  $\log(1/\varepsilon) \geq \log d$ . Let  $m = \left\lceil \frac{\log(1/\varepsilon)}{\log d} \right\rceil$  and let  $n' = \lceil n/m \rceil$ . Plugging into Equation 1, we have constructed already a PRG for length- $n'$  degree- $(d^m)$  permutation branching programs with seed length  $s$ , where

$$\begin{aligned} s &\leq O(\log(d^m) + \log(n/m) \cdot (\log \log n + \log(1/\varepsilon))) \\ &= O\left(\log(1/\varepsilon) + \log\left(\frac{n \log d}{\log(1/\varepsilon)}\right) \cdot (\log \log n + \log(1/\varepsilon))\right) \\ &= O\left(\log\left(\frac{n \log d}{\log(1/\varepsilon)}\right) \cdot (\log \log n + \log(1/\varepsilon))\right), \end{aligned}$$

where the last step uses the assumption  $\varepsilon \geq d^{-n/2}$  which implies  $\log\left(\frac{n \log d}{\log(1/\varepsilon)}\right) \geq 1$ . By Lemma 4.8, that same PRG fools length- $(mn')$  degree- $d$  permutation branching programs. Since  $mn' > n$ , by truncating to the first  $n$  symbols, we get the desired PRG for length- $n$  degree- $d$  permutation branching programs. ◀

## 5 A Random Function is Not a Good PRG

In this section, we prove that a random generator does not fool unbounded-width permutation branching programs, unless the seed length is  $\Omega(n \log d)$ . The proof is based on the following family of exponential-width permutation branching programs.

► **Lemma 5.1.** *Let  $n$  be a multiple of two, and let  $\pi: [d]^{n/2} \rightarrow [d]^{n/2}$  be a permutation. There is a width- $(d^{n/2})$  length- $n$  degree- $d$  permutation branching program  $B$  such that*

$$B(x, y) = 1 \iff y = \pi(x).$$

**Proof.** Let  $\mathbb{Z}_d$  denote the ring of integers modulo  $d$ . We identify the state space  $[d^{n/2}]$  with the space  $\mathbb{Z}_d^{n/2}$ , a  $\mathbb{Z}_d$ -module. Let  $e_1, \dots, e_{n/2} \in \mathbb{Z}_d^{n/2}$  denote the standard “basis vectors,” i.e.,  $e_t$  has a 1 in coordinate  $t$  and 0 in all other coordinates. The transition function  $W_t: \mathbb{Z}_d^{n/2} \times \mathbb{Z}_d \rightarrow \mathbb{Z}_d^{n/2}$  is given by

$$W_t(v, \sigma) = \begin{cases} v + \sigma \cdot e_t & \text{if } t \leq n/2 \\ \pi^{-1}(\pi(v) - \sigma \cdot e_t) & \text{if } t > n/2. \end{cases}$$

These transition functions satisfy the permutation condition, because  $W_t(W_t(v, \sigma), -\sigma) = v$ . The start state of  $B$  is the zero element  $0 \in \mathbb{Z}_d^{n/2}$ , and the accepting state is  $\pi^{-1}(0)$ . By induction, when  $B$  reads an input  $(x, y) \in (\mathbb{Z}_d^{n/2})^2$ , it passes through the state  $x$  in layer  $n/2$ , and ultimately it arrives at the state  $\pi^{-1}(\pi(x) - y)$  in the final layer. Thus,  $B(x, y) = 1 \iff y = \pi(x)$ . ◀

► **Theorem 5.2** (Failure of the Probabilistic Method). *Let  $n$  be a multiple of 2. Let  $s = \left\lfloor \frac{n \log d}{4} \right\rfloor - 1$ , and sample a generator  $G$  uniformly at random from all functions  $G: \{0, 1\}^s \rightarrow [d]^n$ . With probability at least  $3/4$ , there is some length- $n$  degree- $d$  permutation branching program  $B$  such that*

$$\left| \Pr_{\sigma \leftarrow U_{[d]^n}} [B(\sigma) = 1] - \Pr_{x \leftarrow U_{\{0,1\}^s}} [B(G(x)) = 1] \right| = 1 - d^{-n/2}.$$

**Proof.** Let  $G_L, G_R: \{0, 1\}^s \rightarrow [d]^{n/2}$  be the left and right halves of  $G$  respectively, i.e.,  $G(x) = G_L(x) \circ G_R(x)$ . We claim that with high probability,  $G_L$  and  $G_R$  are both injective. Indeed, for each pair of distinct seeds  $x, x' \in \{0, 1\}^s$ , the strings  $G_L(x), G_L(x')$  are independent uniform  $(n/2)$ -symbol strings, so  $\Pr_G[G_L(x) = G_L(x')] = d^{-n/2}$ . The number of pairs  $(x, x')$  is at most  $\binom{2^s}{2} \leq \frac{1}{2} 2^{2s} \leq 2^{-3} d^{n/2}$ , where the last inequality is by our choice of  $s$ . Therefore, by the union bound,  $\Pr_G[G_L \text{ is not injective}] \leq 2^{-3}$ . The same argument applies to  $G_R$  as well, so except with probability  $2 \cdot 2^{-3} = \frac{1}{4}$ ,  $G_L$  and  $G_R$  are both injective. In this case, there exists a permutation  $\pi: [d]^{n/2} \rightarrow [d]^{n/2}$  such that for every seed  $x$ ,  $\pi(G_L(x)) = G_R(x)$ . By Lemma 5.1, there is a length- $n$  degree- $d$  permutation branching program  $B$  such that  $B(y, z) = 1 \iff z = \pi(y)$ . Therefore, for every seed  $x$ ,  $B(G(x)) = 1$ , so  $\Pr_x[B(G(x)) = 1] = 1$ . On the other hand, since  $\pi$  is a permutation,  $\Pr_\sigma[B(\sigma) = 1] = d^{-n/2}$ . ◀

## 6 Seed Length Lower Bound

In this section, we prove our lower bound on the seed length of any PRG for unbounded-width permutation branching programs, showing that our PRG's seed length is near-optimal. Except when  $\varepsilon$  is extremely small, the lower bound is  $\Omega(\log d + \log n \cdot \log(1/\varepsilon))$ .

► **Theorem 6.1.** *Let  $d \geq 2$  and  $n \geq 1$ . Let  $G: \{0, 1\}^s \rightarrow [d]^n$  be an  $\varepsilon$ -PRG for length- $n$  degree- $d$  permutation branching programs of unbounded width, where  $d^{-n/2} \leq \varepsilon \leq 0.49$ . Then*

$$s \geq \Omega \left( \log d + \log \left( \frac{n \log d}{\log(1/\varepsilon)} \right) \cdot \log(1/\varepsilon) \right).$$

The proof of Theorem 6.1 is based on the same family of exponential-width branching programs that we used to prove Theorem 5.2. At an intuitive level, we argue that either the first half of the PRG's output is information-theoretically unpredictable given the second half, or vice versa. After all, if each half is somewhat predictable given the other half, there ought to exist a permutation  $\pi$  such that the pseudorandom string  $(x, y)$  has a noticeable chance (say at least  $2\varepsilon$ ) of satisfying  $\pi(x) = y$ , whereas a truly random string is extremely unlikely to satisfy  $\pi(x) = y$ . It follows that the PRG must use  $\Omega(\log(1/\varepsilon))$  bits of seed above and beyond the seed length for sampling the first half or the second half individually.

To obtain a suitable permutation  $\pi$ , we rely on a version of the Kőnig-Egerváry theorem regarding maximum matchings in bipartite graphs. (In the lemma statement, think of  $p$  as a weight function on the edges of the complete bipartite graph  $K_{N,N}$ , and think of  $\pi$  as identifying a perfect matching.)

► **Lemma 6.2** (Kőnig-Egerváry theorem for fractional edge weights, [21, Theorem 17.1]). *For every integer  $N \geq 1$  and every function  $p: [N] \times [N] \rightarrow [0, \infty)$ , there exist a permutation  $\pi: [N] \rightarrow [N]$  and functions  $q, r: [N] \rightarrow [0, \infty)$  such that*

$$\forall x, y \in [N], p(x, y) \leq q(x) + r(y) \tag{2}$$

and

$$\sum_{x \in [N]} p(x, \pi(x)) = \sum_{x \in [N]} q(x) + \sum_{y \in [N]} r(y). \tag{3}$$



Note that Equations 2 and 3 immediately imply that  $\pi$  maximizes  $\sum_x p(x, \pi(x))$ . A proof of Lemma 6.2 can be found in Schrijver's text [21, Theorem 17.1]. Alternatively, Lemma 6.2 follows from strong linear programming duality and the fact that the integer matching polytope equals the fractional matching polytope.

As outlined previously, we would now like to show that if each of  $X$  and  $Y$  is somewhat predictable given the other, then there is a noticeable chance that  $\pi(X) = Y$ . To rigorously formulate and prove this statement, we use the notion of Shannon entropy.

► **Definition 6.3.** If  $X$  is a discrete random variable, the entropy of  $X$  is

$$H[X] = \mathbb{E}_{x \sim X} \left[ \log \left( \frac{1}{\Pr[X = x]} \right) \right].$$

If  $X$  and  $Y$  are jointly distributed discrete random variables, the joint entropy  $H[X, Y]$  is the entropy of the pair  $(X, Y)$ , and the conditional entropy of  $X$  given  $Y$  is

$$H[X | Y] = \mathbb{E}_{y \sim Y} [H[X | Y = y]] = \mathbb{E}_{\substack{x \sim X \\ y \sim Y}} \left[ \log \left( \frac{1}{\Pr[X = x | Y = y]} \right) \right].$$

► **Lemma 6.4.** Let  $N \geq 1$ , and let  $X$  and  $Y$  be jointly distributed random variables, each taking values in  $[N]$ . There exists a permutation  $\pi: [N] \rightarrow [N]$  such that

$$\Pr[\pi(X) = Y] \geq 2^{-H[X|Y] - H[Y|X]}.$$

Lemma 6.4 bears a resemblance to a well-known fact, which says that if we allow an *arbitrary* function  $\pi$  (not necessarily a permutation), the maximum possible value of  $\Pr[\pi(X) = Y]$  is precisely  $2^{-H_\infty(Y|X)}$ , where  $H_\infty(Y|X)$  is the “average min-entropy” of  $Y$  given  $X$  [7]. Our lemma is an interesting “symmetric” variant.

**Proof.** Let  $\pi, q, r$  be the functions guaranteed by Lemma 6.2 for the function  $p(x, y) = \Pr[(X, Y) = (x, y)]$ . Let  $\mathcal{X} = \text{Supp}(X)$  and  $\mathcal{Y} = \text{Supp}(Y)$ . Then

$$\begin{aligned} \Pr[\pi(X) = Y] &= \sum_{x \in [N]} q(x) + \sum_{y \in [N]} r(y) && \text{(Equation 3)} \\ &\geq \sum_{x \in \mathcal{X}} q(x) + \sum_{y \in \mathcal{Y}} r(y) \\ &= \sum_{\substack{x \in \mathcal{X} \\ y \in \mathcal{Y}}} \Pr[(X, Y) = (x, y)] \cdot \left( \frac{q(x)}{\Pr[X = x]} + \frac{r(y)}{\Pr[Y = y]} \right) \\ &\geq \sum_{\substack{x \in \mathcal{X} \\ y \in \mathcal{Y}}} \Pr[(X, Y) = (x, y)]^2 \cdot \frac{q(x) + r(y)}{\Pr[X = x] \cdot \Pr[Y = y]} \\ &\geq \sum_{\substack{x \in \mathcal{X} \\ y \in \mathcal{Y}}} \frac{\Pr[(X, Y) = (x, y)]^3}{\Pr[X = x] \cdot \Pr[Y = y]} && \text{(Equation 2)} \\ &= \mathbb{E}_{(x, y) \sim (X, Y)} [\Pr[X = x | Y = y] \cdot \Pr[Y = y | X = x]] \\ &\geq 2^{\mathbb{E}_{(x, y) \sim (X, Y)} [\log(\Pr[X = x | Y = y] \cdot \Pr[Y = y | X = x])]} && \text{(Jensen)} \\ &= 2^{-H[X|Y] - H[Y|X]}. \end{aligned}$$

To apply Lemma 6.4 to analyze pseudorandom distributions for permutation branching programs, we will use the standard *chain rule* for Shannon entropy.



▷ **Claim 6.5 (Chain Rule).** If  $X$  and  $Y$  are discrete random variables, then  $H[X, Y] = H[X] + H[Y | X]$ .

► **Lemma 6.6.** Let  $n$  be a multiple of two, let  $X$  and  $Y$  be random variables distributed over  $[d]^{n/2}$ , and let  $\varepsilon \geq d^{-n/2}$ . Assume that for every length- $n$  degree- $d$  permutation branching program  $B$ ,

$$|\Pr[B(U_{[d]^n}) = 1] - \Pr[B(X, Y) = 1]| \leq \varepsilon. \quad (4)$$

Then

$$H[X, Y] \geq \frac{1}{2} \left( H[X] + H[Y] + \log \left( \frac{1}{2\varepsilon} \right) \right).$$

**Proof.** Let  $\pi$  be the permutation of Lemma 6.4. By Lemma 5.1, there is some length- $n$  degree- $d$  permutation branching program  $B$  such that  $B(x, y) = 1 \iff \pi(x) = y$ . Since  $\pi$  is a permutation,  $\Pr[B(U_{[d]^n}) = 1] = d^{-n/2}$ . Therefore, by Equation 4,

$$2^{-H[X|Y] - H[Y|X]} \leq d^{-n/2} + \varepsilon \leq 2\varepsilon.$$

Therefore,

$$\begin{aligned} H[X, Y] &= \frac{1}{2} (H[X] + H[Y | X] + H[Y] + H[X | Y]) && \text{(Chain Rule)} \\ &\geq \frac{1}{2} \left( H[X] + H[Y] + \log \left( \frac{1}{2\varepsilon} \right) \right). && \blacktriangleleft \end{aligned}$$

To complete the proof of Theorem 6.1, we use the following standard fact about Shannon entropy.

▷ **Claim 6.7.** If  $X$  is a discrete random variable and  $f$  is a function, then  $H[f(X)] \leq H[X]$ .

**Proof of Theorem 6.1.** The seed length must be  $\Omega(\log d)$  simply because the program can compute any arbitrary function of its first symbol. For  $i \geq 0$ , let

$$n_i = \left\lceil \frac{\log(1/\varepsilon)}{\log d} \right\rceil \cdot 2^i.$$

We will prove by induction on  $i$  that if a distribution  $X$  over  $[d]^{n_i}$  fools length- $n_i$  degree- $d$  permutation branching programs with error  $\varepsilon$ , then

$$H[X] \geq \frac{i}{2} \cdot \log \left( \frac{1}{2\varepsilon} \right).$$

The base case  $i = 0$  is trivial. For the inductive step, consider a distribution  $(X, Y)$  over strings of length  $n_i$ , where  $|X| = |Y| = n_{i-1}$ . Since a permutation branching program can elect to ignore some of its input symbols,  $X$  and  $Y$  must each individually fool length- $n_{i-1}$  degree- $d$  permutation branching programs with error  $\varepsilon$ . Therefore, by induction,

$$\frac{1}{2} (H[X] + H[Y]) \geq \frac{(i-1)}{2} \cdot \log \left( \frac{1}{2\varepsilon} \right).$$

Furthermore, since  $n_i \geq 2 \log(1/\varepsilon) / \log d$ , we have  $\varepsilon \geq d^{-n_i/2}$ , so we may apply Lemma 6.6 to complete the inductive step.

Now consider

$$i = \left\lfloor \log \left( \frac{n}{\lceil \log(1/\varepsilon) / \log d \rceil} \right) \right\rfloor.$$

Since  $\varepsilon \geq d^{-n/2}$ , we have  $n/2 \leq n_i \leq n$ . Let  $X$  be the truncation of  $G(U_{\{0,1\}^s})$  to the first  $n_i$  symbols. Then

$$s = H[U_{\{0,1\}^s}] \geq H[G(U_{\{0,1\}^s})] \geq H[X] \geq \frac{i}{2} \log \left( \frac{1}{2\varepsilon} \right),$$

where the first two inequalities follow from Claim 6.7. If  $\log(1/\varepsilon) > \log d$ , then  $i = \Omega \left( \log \left( \frac{n \log d}{\log(1/\varepsilon)} \right) \right)$ , so we are done. Meanwhile, if  $\log d = \lambda \cdot \log(1/\varepsilon)$  for some  $\lambda \geq 1$ , then  $i = \lfloor \log n \rfloor$ , so we have shown

$$s \geq \frac{\lfloor \log n \rfloor}{2} \cdot \log \left( \frac{1}{2\varepsilon} \right).$$

We also have

$$s \geq \Omega(\log d) = \Omega(\lambda \cdot \log(1/\varepsilon)) \geq \Omega(\log \lambda \cdot \log(1/\varepsilon)).$$

Combining, we get

$$\begin{aligned} s &\geq \Omega((\log n + \log \lambda) \cdot \log(1/\varepsilon)) \\ &= \Omega(\log(n\lambda) \cdot \log(1/\varepsilon)) \\ &= \Omega \left( \log \left( \frac{n \log d}{\log(1/\varepsilon)} \right) \cdot \log(1/\varepsilon) \right). \end{aligned}$$

◀

## 7 The Optimal Seed Length for Hitting Set Generators

Let  $\mathcal{F}$  be a class of functions  $B: [d]^n \rightarrow \{0,1\}$ . Recall that an  $\varepsilon$ -HSG for  $\mathcal{F}$  is a function  $G: \{0,1\}^s \rightarrow [d]^n$  such that

$$\forall B \in \mathcal{F}, \Pr[B(U_{[d]^n}) = 1] \geq \varepsilon \implies \exists x \in \{0,1\}^s, B(G(x)) = 1.$$

Thus, an HSG is a “one-sided” variant of a PRG.

In this section, we prove that any HSG for polynomial-width permutation branching programs is an HSG for unbounded-width permutation branching programs. As a corollary, we will show that the optimal (nonexplicit) seed length for an HSG for unbounded-width permutation branching programs is  $O(\log(nd/\varepsilon))$ .

► **Theorem 7.1.** *Let  $n$  be a positive integer, let  $G: \{0,1\}^s \rightarrow [d]^n$  be a function, and let  $\varepsilon > 0$ .*

1. *There is a value  $w = O(n/\varepsilon)$  such that if  $G$  is an  $(\varepsilon/2)$ -HSG for width- $w$  length- $n$  ordered branching programs, then  $G$  is an  $\varepsilon$ -HSG for unbounded-width length- $n$  permutation branching programs.*
2. *There is a value  $w = O(n^2/\varepsilon)$  such that if  $G$  is an  $(\varepsilon/2)$ -HSG for width- $w$  length- $n$  permutation branching programs, then  $G$  is an  $\varepsilon$ -HSG for unbounded-width length- $n$  permutation branching programs.*

Item 2 is not necessary for the purpose of establishing the optimal seed length for HSGs for unbounded-width permutation branching programs. We include the proof because we find it interesting.

**Proof.** Let  $B$  be a length- $n$  permutation branching program. We will define a function  $f: [d]^n \rightarrow \{0, 1\}$  such that  $f(x) = 1 \implies B(x) = 1$  and  $\Pr_{x \in [d]^n}[B(x) \neq f(x)] \leq \varepsilon/2$ . Furthermore, we will show that  $f$  can be computed by an ordered branching program of width  $O(n/\varepsilon)$ , as well as by a permutation branching program of width  $O(n^2/\varepsilon)$ .

Think of  $B$  as a directed graph. Let  $V_0, \dots, V_n$  be the layers of the graph. For each vertex  $v$ , let  $p_{\rightarrow v}$  denote the probability that  $B$  passes through  $v$  when it reads a random input. Let  $q = \lceil 2n/\varepsilon \rceil$ . If the width of  $B$  is less than  $q$ , we can just let  $f = B$ , so assume the width of  $B$  is at least  $q$ . For each  $t \in \{0, 1, \dots, n\}$ , define  $S_t$  to be the set of  $q$  vertices  $v \in V_t$  with the largest values of  $p_{\rightarrow v}$ . Let  $f(x) = 1$  if the path through  $B$  described by  $x$  stays within  $S_0, S_1, \dots, S_n$  and ends at the accepting vertex.

Clearly,  $f(x) = 1 \implies B(x) = 1$ . Now consider sampling  $x = (x_1, \dots, x_n) \in [d]^n$  uniformly at random. For each  $t \in [n]$  and each vertex  $v \in V_t$ , let  $B_{v \rightarrow}$  denote the permutation branching program that ignores the first  $t$  symbols of its inputs and then simulates the last  $(n - t)$  layers of  $B$  starting at vertex  $v$ . By the definition of  $S_t$ , each  $v \in V_t \setminus S_t$  satisfies  $p_{\rightarrow v} < 1/q$ . Therefore,

$$\begin{aligned} \Pr_x[B(x) \neq f(x)] &\leq \sum_{t=1}^n \sum_{v \in V_t \setminus S_t} p_{\rightarrow v} \cdot \Pr_x[B_{v \rightarrow}(x) = 1] \\ &< \frac{1}{q} \cdot \sum_{t=1}^n \sum_{v \in V_t \setminus S_t} \Pr_x[B_{v \rightarrow}(x) = 1] \\ &\leq \frac{1}{q} \cdot \sum_{t=1}^n \mathbb{E}_x \left[ \sum_{v \in V_t} B_{v \rightarrow}(x) \right]. \end{aligned}$$

Consider any fixed  $t$  and  $x$ . By the permutation condition, it is possible to work backward from the accepting vertex to find the unique vertex  $v \in V_t$  such that  $B_{v \rightarrow}(x) = 1$ . Therefore,  $\sum_{v \in V_t} B_{v \rightarrow}(x) = 1$ . Thus,

$$\Pr_x[B(x) \neq f(x)] \leq \frac{1}{q} \cdot \sum_{t=1}^n 1 \leq \frac{\varepsilon}{2}.$$

An ordered branching program for  $f$  can be obtained from  $B$  by deleting all the vertices in  $V_t \setminus S_t$  and redirecting all their incoming edges to a new  $\perp$  vertex. All outgoing edges from the  $\perp$  vertex in layer  $t$  point to the  $\perp$  vertex in layer  $t + 1$ , and finally in layer  $n$ , the  $\perp$  vertex is a reject vertex. Clearly, the width of this program is  $q + 1$ .

Now let us define a permutation branching program computing  $f$  of width  $w = q \cdot (n + 1)$ . Let  $w_0$  be the width of  $B$ , and number the states so that  $S_t$  corresponds to  $[q] \subseteq [w_0]$ . Let  $W_t: [w_0] \times [d] \rightarrow [w_0]$  be the transition function of  $B$ . Let  $A_{t,\sigma}$  be the set of  $v \in [q]$  such that  $W_t(v, \sigma) \in [q]$ . By the permutation condition, for each fixed  $t$  and  $\sigma$ , the function  $W_t(\cdot, \sigma)$  is a permutation on  $[w_0]$ . Therefore, there exists a permutation  $\pi_{t,\sigma}: [w_0] \setminus A_{t,\sigma} \rightarrow [w_0] \setminus W_t(A_{t,\sigma}, \sigma)$ .

Let  $\mathbb{Z}_n$  denote the additive group of integers modulo  $n$ , and identify  $[w] = [q] \times \mathbb{Z}_{n+1}$ . The new branching program's transition function  $W'_t: [q] \times \mathbb{Z}_{n+1} \times [d] \rightarrow [q] \times \mathbb{Z}_{n+1}$  is given by

$$W'_t(v, i, \sigma) = \begin{cases} (W_t(v, \sigma), i) & \text{if } v \in A_{t,\sigma} \\ (\pi_{t,\sigma}(v), i + 1) & \text{otherwise.} \end{cases}$$

Clearly, this satisfies the permutation condition. The start state is  $(v_{\text{start}}, 0)$  and the accept state is  $(v_{\text{end}}, 0)$ , where  $v_{\text{start}}$  and  $v_{\text{end}}$  are the start and accept states of  $B$ . If  $f(\sigma) = 1$ , then inductively, when our permutation branching program reads  $\sigma$ , it simulates  $B$  and ultimately

accepts without ever incrementing  $i$ . Conversely, if our permutation branching program accepts  $\sigma$ , then  $i$  must never be incremented. Therefore, when  $B$  reads  $\sigma$ , it stays within the sets  $S_0, \dots, S_n$  and accepts, and hence  $f(\sigma) = 1$ .

Finally, if  $\Pr[B(U_{[d]^n}) = 1] \geq \varepsilon$ , then  $\Pr[f(U_{[d]^n}) = 1] \geq \varepsilon/2$ . Therefore, under either of the two assumptions of the theorem,  $G$  hits  $f$ , and since  $f \leq B$ , this implies that  $G$  hits  $B$  as well.  $\blacktriangleleft$

► **Corollary 7.2.** *For every  $n, d, \varepsilon$ , there exists an  $\varepsilon$ -HSG  $G: \{0, 1\}^s \rightarrow [d]^n$  for unbounded-width length- $n$  degree- $d$  permutation branching programs with seed length  $s = O(\log(nd/\varepsilon))$ .*

**Proof.** It is standard that there exists a nonexplicit  $\varepsilon$ -HSG for width- $w$  length- $n$  degree- $d$  ordered branching programs with seed length  $O(\log(wnd/\varepsilon))$ . (Indeed, a random function is an HSG with these parameters with high probability.)  $\blacktriangleleft$

The next claim shows that the seed length in Corollary 7.2 is optimal.

▷ **Claim 7.3.** Let  $d \geq 2$  and  $n \geq 1$ . Let  $G: \{0, 1\}^s \rightarrow [d]^n$  be an  $\varepsilon$ -HSG for length- $n$  degree- $d$  permutation branching programs of unbounded width, where  $d^{-n} \leq \varepsilon \leq 1/3$ . Then  $s \geq \Omega(\log(nd/\varepsilon))$ .

*Proof sketch.* The seed length needs to be at least  $\Omega(\log d)$  because the program can compute any function of the first input symbol. The seed length needs to be at least  $\Omega(\log(1/\varepsilon))$  because unbounded-width permutation branching programs can check whether a prefix of the input is equal to a fixed arbitrary string. Finally, let  $G: \{0, 1\}^s \rightarrow [d]^n$  with  $s < \log n$ ; we will show that  $G$  is not a  $(1/3)$ -HSG for degree- $d$  permutation branching programs. Let  $b: [d] \rightarrow \mathbb{F}_2$  be as close to balanced as possible. Since  $2^s < n$ , there is some nonzero vector  $z \in \mathbb{F}_2^n$  such that for every seed  $x$ ,

$$\sum_{i=1}^n z_i \cdot b(G(x)_i) = 0.$$

The function  $B(x) = \sum_{i=1}^n z_i \cdot b(x_i)$  can be computed by a width-2 degree- $d$  permutation branching program, and  $\Pr_x[B(x) = 1] \geq 1/3$ .  $\triangleleft$

## 8 Directions for Further Research

The obvious challenge is to obtain optimal PRGs for unbounded-width permutation branching programs in the large-error regime. We conjecture that our seed length lower bound is tight, i.e., there is a PRG construction that eliminates the  $\log \log n$  factor from our PRG's seed length.

We showed that there is a nonexplicit HSG with seed length  $O(\log(n/\varepsilon))$  for unbounded-width permutation branching programs. A natural problem is to match the seed length with an explicit construction. In the constant-width case, Braverman et al. [2] presented a simple HSG for the more general model of regular branching programs with seed length  $O(\log n)$ , independent of  $\varepsilon$ .

We wonder what PRGs can be constructed for the more challenging model of *arbitrary-order* permutation branching programs. Reingold, Steinke, and Vadhan [17] and Chattopadhyay et al. [3] have constructed PRGs for the small-width case. By using one generator for large  $\varepsilon$  and the other for small  $\varepsilon$ , one can achieve seed length  $\tilde{O}(\log n \cdot \log(1/\varepsilon))$  when the width is a constant. For the unbounded-width case, explicit constructions or bounds for nonexplicit PRGs would be interesting.

Finally, we wonder whether our results can be generalized to the case of unbounded-width regular branching programs. Our HSG existence proof (Theorem 7.1 and Corollary 7.2) does generalize to the regular case<sup>3</sup>, but the PRG situation is unclear.

---

## References

- 1 AmirMahdi Ahmadinejad, Jonathan Kelner, Jack Murtagh, John Peebles, Aaron Sidford, and Salil Vadhan. High-precision estimation of random walks in small space. In *Proceedings of the 61st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2020. To appear.
- 2 Mark Braverman, Anup Rao, Ran Raz, and Amir Yehudayoff. Pseudorandom generators for regular branching programs. *SIAM Journal on Computing*, 43(3):973–986, 2014. doi:10.1137/120875673.
- 3 Eshan Chattopadhyay, Pooya Hatami, Kaave Hosseini, and Shachar Lovett. Pseudorandom generators from polarizing random walks. *Theory of Computing. An Open Access Journal*, 15:Paper No. 10, 26, 2019. doi:10.4086/toc.2019.v015a010.
- 4 Michael B. Cohen, Jonathan Kelner, John Peebles, Richard Peng, Anup B. Rao, Aaron Sidford, and Adrian Vladu. Almost-linear-time algorithms for Markov chains and new spectral primitives for directed graphs. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 410–419. ACM, New York, 2017. doi:10.1145/3055399.3055463.
- 5 Anindya De. Pseudorandomness for permutation and regular branching programs. In *26th Annual IEEE Conference on Computational Complexity*, pages 221–231. IEEE Computer Soc., Los Alamitos, CA, 2011. doi:10.1109/CCC.2011.23.
- 6 Anindya De, Omid Etesami, Luca Trevisan, and Madhur Tulsiani. Improved pseudorandom generators for depth 2 circuits. In *Approximation, randomization, and combinatorial optimization*, volume 6302 of *Lecture Notes in Computer Science*, pages 504–517. Springer, Berlin, 2010. doi:10.1007/978-3-642-15369-3\_38.
- 7 Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam Smith. Fuzzy extractors: how to generate strong keys from biometrics and other noisy data. *SIAM Journal on Computing*, 38(1):97–139, 2008. doi:10.1137/060651380.
- 8 Shlomo Hoory and Avi Wigderson. Universal traversal sequences for expander graphs. *Information Processing Letters*, 46(2):67–69, 1993. doi:10.1016/0020-0190(93)90199-J.
- 9 Russell Impagliazzo, Noam Nisan, and Avi Wigderson. Pseudorandomness for network algorithms. In *Proceedings of the 26th Annual ACM Symposium on Theory of Computing (STOC)*, page 356–364, New York, NY, USA, 1994. Association for Computing Machinery. doi:10.1145/195058.195190.
- 10 Michal Koucký, Prajakta Nimbhorkar, and Pavel Pudlák. Pseudorandom generators for group products. In *Proceedings of the 43rd ACM Symposium on Theory of Computing (STOC)*, pages 263–272. ACM, New York, 2011. doi:10.1145/1993636.1993672.
- 11 M. Luby and B. Veličković. On deterministic approximation of DNF. *Algorithmica*, 16(4-5):415–433, 1996. doi:10.1007/s004539900054.
- 12 Raghu Meka and David Zuckerman. Pseudorandom generators for polynomial threshold functions. *SIAM Journal on Computing*, 42(3):1275–1301, 2013. doi:10.1137/100811623.
- 13 Jack Murtagh, Omer Reingold, Aaron Sidford, and Salil Vadhan. Derandomization beyond connectivity: undirected Laplacian systems in nearly logarithmic space. In *Proceedings of the 58th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 801–812. IEEE Computer Soc., Los Alamitos, CA, 2017. doi:10.1109/FOCS.2017.79.

---

<sup>3</sup> In the regular case, we have  $\mathbb{E}_x[\sum_{v \in V_t} B_{v \rightarrow}(x)] = \sum_{v \in V_t} \Pr_x[B_{v \rightarrow}(x) = 1] = 1$ , where the last equality can be proven by backward induction on  $t$ .

- 14 Jack Murtagh, Omer Reingold, Aaron Sidford, and Salil Vadhan. Deterministic Approximation of Random Walks in Small Space. In Dimitris Achlioptas and László A. Végh, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2019)*, volume 145 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 42:1–42:22, Dagstuhl, Germany, 2019. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.APPROX-RANDOM.2019.42.
- 15 Noam Nisan. Pseudorandom generators for space-bounded computation. *Combinatorica*, 12(4):449–461, 1992. doi:10.1007/BF01305237.
- 16 Omer Reingold. Undirected connectivity in log-space. *Journal of the ACM*, 55(4):Art. 17, 24, 2008. doi:10.1145/1391289.1391291.
- 17 Omer Reingold, Thomas Steinke, and Salil Vadhan. Pseudorandomness for regular branching programs via Fourier analysis. In *Approximation, randomization, and combinatorial optimization*, volume 8096 of *Lecture Notes in Computer Science*, pages 655–670. Springer, Heidelberg, 2013. doi:10.1007/978-3-642-40328-6\_45.
- 18 Omer Reingold, Luca Trevisan, and Salil Vadhan. Pseudorandom walks on regular digraphs and the  $\mathbf{RL}$  vs.  $\mathbf{L}$  problem. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing (STOC)*, pages 457–466. ACM, New York, 2006. doi:10.1145/1132516.1132583.
- 19 Omer Reingold, Salil Vadhan, and Avi Wigderson. Entropy waves, the zig-zag graph product, and new constant-degree expanders. *Annals of Mathematics. Second Series*, 155(1):157–187, 2002. doi:10.2307/3062153.
- 20 Eyal Rozenman and Salil Vadhan. Derandomized squaring of graphs. In *Approximation, randomization and combinatorial optimization*, volume 3624 of *Lecture Notes in Computer Science*, pages 436–447. Springer, Berlin, 2005. doi:10.1007/11538462\_37.
- 21 Alexander Schrijver. *Combinatorial optimization. Polyhedra and efficiency. Vol. A*, volume 24 of *Algorithms and Combinatorics*. Springer-Verlag, Berlin, 2003. Paths, flows, matchings, Chapters 1–38.
- 22 Thomas Steinke. Pseudorandomness for permutation branching programs without the group theory. ECCC preprint TR12-083, 2012.

# Pipeline Interventions

**Eshwar Ram Arunachaleswaran**

University of Pennsylvania, Philadelphia, PA, USA  
eshwar@seas.upenn.edu

**Sampath Kannan**

University of Pennsylvania, Philadelphia, PA, USA  
kannan@cis.upenn.edu

**Aaron Roth**

University of Pennsylvania, Philadelphia, PA, USA  
aaro@cis.upenn.edu

**Juba Ziani**

University of Pennsylvania, Philadelphia, PA, USA  
jziani@seas.upenn.edu

---

## Abstract

We introduce the *pipeline intervention* problem, defined by a layered directed acyclic graph and a set of stochastic matrices governing transitions between successive layers. The graph is a stylized model for how people from different populations are presented opportunities, eventually leading to some reward. In our model, individuals are born into an initial position (i.e. some node in the first layer of the graph) according to a fixed probability distribution, and then stochastically progress through the graph according to the transition matrices, until they reach a node in the final layer of the graph; each node in the final layer has a *reward* associated with it. The pipeline intervention problem asks how to best make costly changes to the transition matrices governing people's stochastic transitions through the graph, subject to a budget constraint. We consider two objectives: social welfare maximization, and a fairness-motivated maximin objective that seeks to maximize the value to the population (starting node) with the *least* expected value. We consider two variants of the maximin objective that turn out to be distinct, depending on whether we demand a deterministic solution or allow randomization. For each objective, we give an efficient approximation algorithm (an additive FPTAS) for constant width networks. We also tightly characterize the “price of fairness” in our setting: the ratio between the highest achievable social welfare and the social welfare consistent with a maximin optimal solution. Finally we show that for polynomial width networks, even approximating the maximin objective to any constant factor is NP hard, even for networks with constant depth. This shows that the restriction on the width in our positive results is essential.

**2012 ACM Subject Classification** Theory of computation → Algorithmic game theory

**Keywords and phrases** Interventions for fairness, fairness in navigating life paths, social welfare, maximin welfare, budget-constrained optimization, hardness of approximation

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.8

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/2002.06592> [1].

**Funding** *Eshwar Ram Arunachaleswaran*: Supported in part by NSF grant CCF-1763307.

*Sampath Kannan*: Supported in part by NSF grants CCF-1763307 and CCF-1733794.

*Aaron Roth*: Supported in part by NSF grant CCF-1763307 and a grant from the Simons Foundation.

*Juba Ziani*: Supported in part by NSF grant CCF-1763307 and by the Warren Center.



© Eshwar Ram Arunachaleswaran, Sampath Kannan, Aaron Roth, and Juba Ziani;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 8; pp. 8:1–8:20



Leibniz International Proceedings in Informatics

LIPIcs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



## 1 Introduction

Inequality can be difficult to correct by the time it manifests itself in consequential domains. For example, faculty in computer science departments are disproportionately male (Way et al. [21]), and although the reasons for this are varied and complex, it seems difficult to correct *only* by intervening in the process of faculty hiring (although the solution likely involves some intervention at this stage). The problem is that interventions at the final stage of a long pipeline may not be enough (or the best way) to address inequities that compound starting from earlier stages in the pipeline such as graduate school, college, high school, enrichment programs, all the way back to birth circumstances. Because each stage of, for example, employment pipelines feeds into the next, interventions that are isolated to any one stage can have difficulty controlling effects on final outcomes – and although in practice it is difficult to fully understand such a system, we would ideally like to design proposed interventions at a system-wide level, rather than myopically.

Thus motivated, we study an optimization problem within a stylized (and highly simplified) model of such a pipeline. Our model is a layered directed acyclic graph. The vertices in the first layer represent a coarse partitioning of possible birth circumstances into a small number of types – each vertex representing one of these types. There is a probability vector over these vertices and individuals are “born” into some vertex with these probabilities. The graph represents a Markov process that determines how individuals progress through the pipeline. From every vertex there is a stochastic transition matrix specifying the probability that an individual will progress to each vertex in the next layer of the pipeline. We might imagine, for example, that the proportion of children that enroll in each of several elementary schools (the second layer of such a pipeline) varies according to the neighborhood that they are raised in (the first layer). The proportion of children that then go on to enroll in each of several high schools may then vary according to the elementary school they attend, and so on. Finally, vertices at the last layer of the pipeline are associated with payoffs. One may then calculate the expected payoff of an individual as a function of their initial position. These payoffs may vary widely depending on this position.

We are concerned with the problem of how best to invest limited resources so as to *modify* the transition matrices governing different layers of this pipeline to achieve some goal. In the main body of the paper, we focus on a stylized model where the costs of modifying transition matrices are linear, for simplicity of exposition; we extend our results to more complex and realistic cost functions in the Appendix. We consider two goals: the first is simply maximizing social welfare – the expected payoff for an individual chosen according to the given probability vector for the first layer. Although this is a natural objective, it can easily lead to solutions that are “unfair” in the sense that they will prioritize investments that lead to improvements for majority populations over minority populations, simply because majority populations, by their sheer numbers, contribute more to social welfare. The second goal we study is therefore to maximize the *minimum* expected payoff of individuals, where the minimum is taken over all of the initial positions, i.e., layer 1 vertices. This “maximin” objective is a standard fairness-motivated objective in allocation problems [See Barman and Krishnamurthy [2], Procaccia and Wang [19], Budish [5]]. In fact, we study two different variants of this objective, that can be distinguished by the timing with which one wants to evaluate fairness. The *ex-ante* maximin objective asks for a *distribution* over budget-feasible modifications of the transition matrices, that maximize the minimum expected payoff over all initial positions. The *ex-post* maximin objective asks for a single (i.e. deterministic) budget-feasible modification to the transition matrices. Because the problem we study is non-convex, these two goals are distinct – which is preferred depends on *when* one wants to evaluate the fairness of a solution: before or after the randomization.



## 1.1 Overview of Our Results

Briefly, our main contributions are the following:

1. We define and formalize the *pipeline intervention problem* with the social welfare, ex-ante maximin, and ex-post maximin objectives. We also prove a separation between the ex-post and ex-ante maximin solutions.
2. We give an additive fully polynomial-time approximation scheme (FPTAS) for both the social welfare and ex-post maximin objectives for networks of constant width (but arbitrarily long depth).
3. We give an efficient reduction from the ex-ante maximin objective problem to the ex-post maximin objective problem via equilibrium computation in two-player zero-sum games. Combined with our results from 2, this yields an additive FPTAS for the ex-ante maximin objective problem for constant width networks as well.
4. We define and prove tight bounds on the “price of fairness”, which compares the optimal social welfare that can be achieved with a given budget to the social welfare of ex-post maximin optimal solutions.
5. Finally, we show that the pipeline intervention problem is NP hard even to approximate in the general case when the width  $w$  is not bounded – and hence that our efficient approximation algorithms cannot be extended to the general case (or even the case of constant depth, polynomial width networks).

## 1.2 Related Work

There is an enormous literature in “algorithmic fairness” that has emerged over the last several years, that we cannot exhaustively summarize here – but see Chouldechova and Roth [6] for a recent survey. Most of this literature is focused on the myopic effects of a single intervention, but what is more conceptually related to our paper is work focusing on the longer-term effects of algorithmic interventions.

Dwork and Ilvento [9] and Bower et. al. [4] study the effects of imposing fairness constraints on machine learning algorithms that might be composed together in various ways to reach an eventual outcome. They show that generally fairness constraints imposed on constituent algorithms in a pipeline or other composition do not guarantee that the same fairness constraints will hold on the entire mechanism as a whole. (They also study conditions under which fairness guarantees *are* well behaved under composition). Two recent papers (Liu et al. [15], Mouzannar et al. [17]) study parametric models by which classification interventions in an earlier stage can have effects on the data distribution at later stages, and show that for many commonly studied fairness constraints, their effects can either be positive or negative in the long term, depending on the functional form of the relationship between classification decisions and changes in the agent type distribution.

There is also a substantial body of work studying game theoretic models for how interventions affect “fairness” goals. This work dates back to Coate and Loury [7], Foster and Vohra [10] in the economics literature, who propose game theoretic models to rationalize how unequal outcomes might emerge despite two populations being symmetrically situated. More recently, in the computer science literature, several papers consider more complicated models that are similar in spirit to Coate and Loury [7], Foster and Vohra [10]. Hu and Chen [12] propose a two-stage model of a labor market with a “temporary” (i.e. internship) and “permanent” stage, and study the equilibrium effects of imposing a fairness constraint on the temporary stage. Liu et al. [16] consider a model of the labor market with higher dimensional signals, and study equilibrium effects of subsidy interventions which can lessen

the cost of exerting effort. Kannan et al. [14] study the effects of admissions policies on a two-stage model of education and employment, in which a downstream employer makes rational decisions. Jung et al. [13] study a model of criminal justice in which crime rates are responsive to the classifiers used to determine criminal guilt, and study which fairness constraints are consistent with the goal of minimizing crime.

## 2 Model

The *pipeline intervention* problem is defined by a layered directed acyclic graph  $G = (V, E)$ , where  $V$  is the set of vertices (or nodes), and  $E$  is the set of edges. The vertices are partitioned into  $k$  layers  $L_1, L_2, \dots, L_k$ , each consisting of  $w$  vertices. We say that  $w$  is the *width* of the graph. For every  $t \in [k - 1]$ , there is a directed edge from every  $u \in L_t$  to every  $v \in L_{t+1}$ ; the graph contains no other edge. In turn, every path from layer  $L_1$  to layer  $L_k$  must go through exactly one vertex in each layer  $L_2, \dots, L_{k-1}$  in this order. Intuitively, such a layered graph represents a pipeline, in which individuals start at initial positions in layer 1, and transition through the graph to final positions in layer  $k$ , stochastically according to transition matrices which we define next. This layered model can be used to abstractly represent real-life pipelines; such a pipeline, that has received attention in previous work (e.g. Kannan et al. [14]), is the education and job market one. Nodes in the initial layer represent a coarse partitioning of the population based on family income levels and educational background. The second layer could represent pre-K experience. For example, one could have 3 nodes in the second layer representing no pre-K, Headstart, and private pre-K. See for example Barnum [3] for a general discussion of as well as pointers to recent studies on the efficacy of Headstart programs. At the next level or two, nodes can represent different qualities of K-12 schools, based on a coarse partitioning of their performance under one of several widely-available metrics, such as the ones provided by U.S. News [20], Niche [18].

The layer after that could be a coarse partitioning where nodes represent, for example, no college, technical or vocational school, and 2 and 4-year colleges coarsely grouped together based on perceived quality according to one of several college rankings. A subsequent layer could encode the details of a student's performance in college, such as their major and GPA, again under a coarse bucketing. The last layer, with numerical rewards could represent different types of employment with rewards determined by starting salaries and prospects for advancement.

In a more accurate model, we might perhaps condition the probability of transition from node  $u$  in layer  $i$  to node  $v$  in layer  $i + 1$  on the entire path taken by an individual leading up to node  $u$ . However, for mathematical tractability, we make the simplifying assumption that the process is Markovian, and this transition probability from  $u$  is independent of prior history.

Let  $\mathcal{M}$  be the set of left stochastic matrices in  $\mathbb{R}^{w \times w}$ : i.e.,  $M \in \mathcal{M}$  if and only if for all  $j \in [w]$ ,  $\sum_{i \in [w]} M(i, j) = 1$ , and for all  $(i, j) \in [w]^2$ ,  $M(i, j) \geq 0$ . Let  $\mathcal{D} \triangleq \{x \in [0, 1]^w : \sum_{k=1}^w x(k) = 1\}$  be the set of probability distributions over  $[w]$ . An instance of the pipeline intervention problem is defined by three elements:

1. A set of initial transition matrices  $M_t^0 \in \mathcal{M}$  between layers  $L_t$  and  $L_{t+1}$ , for all  $t \in [k - 1]$ , such that for all  $u \in L_t$ ,  $v \in L_{t+1}$ ,  $M_t^0(v, u)$  denotes the probability of transitioning from node  $u$  to node  $v$ . Note that we will multiply any input distribution to the right of any transition matrix we use in the paper.

2. An input distribution  $D_1$  over the vertices in layer 1, where  $D_1(u)$  denotes the fraction of the population that starts at  $u$  in  $L_1$  as their initial position. Without loss of generality we assume  $D_1(u) > 0$  for all initial positions  $u \in L_1$ .
3. Finally, a reward  $R(v) \geq 0$  corresponding to each vertex  $v \in L_k$  in the final layer. We let  $R = (v)_{v \in L_k}^\top$  denote the vector of all rewards on layer  $k$ . We assume without loss of generality that the rewards on any two vertices in the final layer are distinct: for all  $v, v' \in L_k$ ,  $R(v) \neq R(v')$ . We can also assume without loss of generality (up to renaming) that  $R(1) > \dots > R(w)$ .

In our model, each vertex  $u$  in the *starting layer*  $L_1$  represents the initial position of some population; abusing notation, we refer to this population also as  $u$ . An individual in population  $u$  transitions to a node in layer  $L_2$ , then a node in layer  $L_3$ , up until they reach a node  $v$  in *destination layer*  $L_k$ , and obtains a reward of  $R(v)$ , with probability given by the transition matrices  $M_1^0$  to  $M_{k-1}^0$ . The expected reward of an individual from population  $u$  is therefore given by  $R^\top M_{k-1} \dots M_1 e_u$ , where  $e_u$  represents the  $w$ -dimensional standard basis vector corresponding to index  $u$ . The aim of the pipeline intervention problem is to *modify* the transition matrices between pairs of adjacent layers so as to improve these expected rewards in some way (we study several objectives) given a finite resource constraint.

We will take the point of view of a centralized designer, who can invest money into modifying the transition matrices between layers. We assume some edges can be modified, while some edges cannot; the edges that can be modified are called *malleable*, and the edges that cannot be modified are called *non-malleable*. We denote the set of malleable edges between layers  $t$  and  $t+1$  by  $E_{mal}^t$  and the set of non-malleable edges by  $\overline{E_{mal}^t}$  its complement. Further, we assume that modifying these transitions matrices comes at a cost, and that on a given layer  $t$ , the cost of transforming  $M_t^0$  to some alternative  $M_t \in \mathcal{M}$  is given by:

$$c(M_t, M_t^0) \triangleq \sum_{(i,j) \in [w]^2} |M_t(i,j) - M_t^0(i,j)|.$$

► **Remark 1.** A critique of such cost functions is that they may not be rich enough to model the cost of improving transitions and opportunities between different stages of, say, the education pipeline.

To address this, we note that while we focus on these simple cost functions in the main body of the paper for simplicity of exposition, our algorithmic results (of Sections 4, 5 and 6) extend to more general and possibly more realistic cost functions – so long as they are convex and increase at least linearly as the distance  $\sum_{(i,j) \in [w]^2} |M_t(i,j) - M_t^0(i,j)|$  between modified transition matrix  $M_t$  and initial transition matrix  $M_t^0$  increases. We discuss this extension in more detail in the full version [1].

This extension allows us to model more realistic situations such as those where the cost functions are not linear, but also those where different edges have different costs – as motivated by the fact that real-life interventions often become more expensive the later they happen.

The designer has a total budget of  $B$ , and can select target transition matrices  $(M_1, \dots, M_{k-1})$  so long as the cost of modifying the initial transition matrices to his targets does not exceed his budget, and only malleable edges have been modified. That is, he must select target transition matrices subject to the constraint:

$$\sum_{t=1}^{k-1} c(M_t, M_t^0) \leq B.$$

We let

$$\begin{aligned} & \mathcal{F}(B, M_1^0, \dots, M_{k-1}^0) \\ &= \left\{ (M_1, \dots, M_{k-1}) : \sum_{t=1}^{k-1} c(M_t, M_t^0) \leq B, M_t \in \mathcal{M} \forall t, M_t(i, j) = M_t^0(i, j) \forall (i, j) \in \overline{E_t^{mal}} \right\} \end{aligned}$$

be the set of feasible sets of transition matrices, given initial matrices  $M_1^0, \dots, M_{k-1}^0$  and budget  $B$ . We will consider several objectives that we may wish to optimize. The first is simply to maximize the overall social welfare (i.e. the expected reward of an individual chosen according to  $D_1$ ), which is given by

$$W(M_1, \dots, M_{k-1}) \triangleq R^\top M_{k-1} \dots M_1 D_1.$$

The second objective aims to compute a “fair” outcome in the sense that it evaluates a solution according to the expected payoff of the worst-off members of society (here interpreted as individuals starting at the pessimal initial position), rather than according to the average. This is the classic maximin objective. It turns out that there are two distinct variants of this problem, depending on whether one wishes to allow randomized solutions (i.e. distributions over matrices) or not. We will elaborate on this distinction in the next section, but in the deterministic variant we wish to optimize

$$\min_{j \in [w]} R^\top M_{k-1} \dots M_1 e_j,$$

where  $e_j \in \mathbb{R}^w$  is the unit vector with  $e_j(j) = 1$ , and  $e_j(i) = 0$  for all  $i \neq j$ .

► **Remark 2.** We have assumed that each layer has *exactly*  $w$  vertices. In fact, all of our results generalize to the case in which each layer has  $\leq w$  vertices.

## 2.1 Optimization Problems of Interest

In this paper, we will provide algorithms to solve the following three optimization problems. We note at the outset that these optimization problems are non-convex, due to the fact that our objective values are not convex for  $k \geq 2$ . Hence we should not expect efficient algorithms in the fully general setting; we will give efficient algorithms for networks of constant width  $w$  (i.e. algorithms whose running time is polynomial in the depth of the network  $k$ ), and show that outside of this class, the problem is NP hard even to approximate.

- **Social welfare maximization** The first optimization problem we aim to solve is that of maximizing the social welfare of our network, under our budget constraint:

$$\begin{aligned} OPT_{SW} &= \max_{M_1, \dots, M_{k-1}} R^\top M_{k-1} \dots M_1 D_1^0 \\ \text{s.t. } & (M_1, \dots, M_{k-1}) \in \mathcal{F}(B, M_1^0, \dots, M_{k-1}^0) \end{aligned} \quad (1)$$

- **Ex-post maximin problem** The second optimization problem aims to maximize the minimum expected reward that a population can obtain, where the minimum is taken over all initial positions:

$$\begin{aligned} OPT_{MM} &= \max_{M_1, \dots, M_{k-1}} \min_{j \in [w]} R^\top M_{k-1} \dots M_1 e_j \\ \text{s.t. } & (M_1, \dots, M_{k-1}) \in \mathcal{F}(B, M_1^0, \dots, M_{k-1}^0) \end{aligned} \quad (2)$$

- **Ex-ante maximin problem** The third optimization problem has the same objective as Program 2, but allows randomization over sets of transition matrices that satisfy the budget constraint. Note that the budget constraint must be satisfied *ex-post*, for *any realization* of the set of transition matrices. To define this optimization problem, we let  $\Delta\mathcal{F}(B, M_1^0, \dots, M_{k-1}^0)$  the set of probability distributions with support  $\mathcal{F}(B, M_1^0, \dots, M_{k-1}^0)$ . The optimization program is given by:

$$\begin{aligned} OPT_{RMM} = \max_{\Delta M} \quad & \min_{j \in [w]} R^\top \mathbb{E}_{M \sim \Delta M} [M_{k-1} \dots M_1] e_j \\ \text{s.t.} \quad & \Delta M \in \Delta\mathcal{F}(B, M_1^0, \dots, M_{k-1}^0), \end{aligned} \quad (3)$$

where the expectation is taken over the randomness of distribution  $\Delta M$ . Note that where Program 3 can be viewed as optimizing an *ex-ante* notion of fairness, in which we are evaluated on the minimum expected value of individuals starting at any initial position, *before the coins of  $\Delta M$  are flipped*. In contrast, Program 2 evaluates the minimum expected value of individuals starting at any initial position for an already established set of transition matrices.

► **Remark 3.** Programs (1), (2) and (3) all have solutions, and as such the use of maxima instead of suprema is well defined. To see this, first note that the feasible sets are non-empty since  $(M_1^0, \dots, M_{k-1}^0) \in \mathcal{F}(B, M_1^0, \dots, M_{k-1}^0)$  for all  $B \geq 0$ . For Program (1), the existence of a maximum is an immediate consequence of the fact that the objective function is continuous in  $(M_1, \dots, M_{k-1})$  and  $\mathcal{F}$  and  $\mathcal{M}$  are compact sets. For Program (2), note that no solution can have  $R^\top M_{k-1} \dots M_1 e_j \geq \|R\|_\infty$  for any  $j$ , as  $M_{k-1} \dots M_1 e_j$  is a probability distribution. Hence, we can rewrite the program as

$$\begin{aligned} \max_{v, M_1, \dots, M_{k-1}} \quad & v \\ \text{s.t.} \quad & 0 \leq v \leq \|R\|_\infty, \\ & R^\top M_{k-1} \dots M_1 e_j \geq v \quad \forall j \in [w], \\ & (M_1, \dots, M_{k-1}) \in \mathcal{F}(B, M_1^0, \dots, M_{k-1}^0). \end{aligned}$$

This is an optimization problem with a continuous objective function over a compact set, so it admits a solution. A similar argument follows for Program (3).

### 3 Algorithmic Preliminaries

Our paper uses a dynamic programming approach for solving programs (1) and (2). (Our solution to program (3) is a game-theoretic reduction to our solution to program (2)). Our algorithms will search over possible input distributions in  $\mathcal{D}$  starting from layer  $L_t$  for all  $t \in \{2, \dots, k-2\}$ , and over possible ways of splitting the total budget  $B$  and allocating budget  $B_t$  to the transition from layer  $L_t$  to layer  $L_{t+1}$ , for all  $t \in [k-1]$ . To do so, we will need to discretize both the budget space  $[0, B]$  and the probability space  $\mathcal{D}$ .

#### 3.1 Cost of Discretizing the Budget

To discretize the budget space, we define  $\mathcal{B}(\varepsilon) = \{k\varepsilon, \forall k \in \mathcal{N}\}$  to be the set of numbers on the real line that are multiples of  $\varepsilon$ . We consider the following discretized version of Program 1:

$$\begin{aligned}
OPT_{SW}^\varepsilon &= \max_{M_1, \dots, M_{k-1}} R^\top M_{k-1} \dots M_1 D_1 \\
\text{s.t. } &c(M_t, M_t^0) \leq B_t \quad \forall t \in [k-1] \\
&B_t \in \mathcal{B}(\varepsilon) \quad \forall t \in [k-1], \quad \sum_{t=1}^{k-1} B_t \leq B \\
&M_t(i, j) = M_t^0(i, j) \quad \forall (i, j) \in \overline{E_t^{mal}}, \quad M_t \in \mathcal{M} \quad \forall t
\end{aligned} \tag{4}$$

Program 2 needs an analogous modification. (We do not need to explicitly consider Program (3), since our solution for this one will be a reduction to our solution to Program (2).)

We show that this discretization does not affect the optimal value of our problems by much:

► **Claim 4.** There exists a feasible solution  $(M_1^\varepsilon, \dots, M_{k-1}^\varepsilon)$  to Program (4) (resp. for the analogous modification of Program ) with objective value at least  $OPT_{SW} - (k-1)\varepsilon \|R\|_\infty$  (resp.  $OPT_{MM} - (k-1)\varepsilon \|R\|_\infty$ ).

We provide a brief proof sketch below, and defer the full proof to the full version [1].

*Proof Sketch.* We prove this result by constructing transition matrices  $M_t^\varepsilon$  that use roughly  $\varepsilon$  budget less than  $M_t^*$ . We show that we can do so so as to only lose welfare of the order of  $\varepsilon$  in each of the  $k-1$  layer transitions we consider, and that this loss composes additively.  $\triangleleft$

► **Definition 5.** Let  $K \subseteq [0, 1]^w$ . We call a subset  $S$  of  $K$  an  $\varepsilon$ -net for  $K$  with respect to the  $\ell_1$ -norm if and only if for every  $D \in K$ , there exists  $D' \in S$  such that

$$\|D - D'\|_1 \leq \varepsilon.$$

► **Claim 6** ( $\varepsilon$ -nets in  $\ell_1$ -distance for  $\mathcal{D}$ ). Take  $\varepsilon > 0$ . There exists an  $\varepsilon$ -net  $\mathcal{D}(\varepsilon)$  of  $\mathcal{D}$  with respect to the  $\ell_1$ -norm that has size  $\left(\frac{1}{\varepsilon}\right)^w$ .

This is a standard proof, which can be found in the full version of this paper [1].

## 4 Social Welfare Maximization

We want to solve the following optimization problem:

$$\begin{aligned}
&\max_{M_1, \dots, M_{k-1}} R^\top M_{k-1} \dots M_1 D_1 \\
\text{s.t. } &\sum_{t=1}^{k-1} c(M_t, M_t^0) \leq B, \\
&M_t \in \mathcal{M} \quad \forall t \in [k-1],
\end{aligned} \tag{5}$$

### 4.1 A Dynamic Programming Algorithm for Social Welfare Maximization

In this section, we describe a dynamic programming algorithm for approximately solving the problem above on long skinny networks. The algorithm will run in polynomial time when the width  $w$  of the network is small; its running time is polynomial in the depth  $k$  of the network, but exponential in the width  $w$ . The formal description can be found in the full version. Our

algorithm works backwards, starting from the final transition matrix from layer  $L_{k-1}$  to  $L_k$ . It builds up the solutions to sub-problems parameterized by three parameters – a layer  $t$ , a starting distribution over the vertices in layer  $t$ , and a budget  $B_{\geq t}$  that can be used at layers  $\geq t$ . For each sub-problem, it computes an approximately welfare-optimal solution. Once all of these sub-problems have been solved, the optimal solution to the original problem can be read off from the “sub-problem” in which  $t = 1$ , the starting distribution is the distribution on initial positions, and  $B_{\geq 1} = B$ . Here is the informal description of the algorithm:

1. For  $t$  going backwards from  $k - 1$  to 1, the algorithm does the following exploration over budget splits and probability distributions  $D_t, D_{t+1} \in \mathcal{D}(\varepsilon)$  (an  $\varepsilon$ -net for the  $w$ -dimensional simplex in  $\ell_1$  norm) on  $L_t$ :
  - a. The algorithm explores all discretized splits of a budget  $B_{\geq t}$  to be used for layers  $t$  to  $k - 1$  into a budget  $B_t$  to expend on layer  $t$  and a budget  $B_{\geq t+1}$  to expend on the remaining layers  $t + 1$  to  $k - 1$ , as well as all choices of target output probability distribution  $D_{t+1} \in \mathcal{D}(\varepsilon)$  on layer  $L_{t+1}$  and the starting probability distribution  $D_t \in \mathcal{D}(\varepsilon)$ . Informally, we can think of these “target” and “initial” probability distributions as guesses for what the distribution on vertices in layer  $t + 1$  and layer  $t$  look like in the optimal solution. Recall that for each  $D_{t+1}$  and  $B_{\geq t+1}$ , our algorithm has already computed a near-optimal solution for a smaller sub-problem, which we will utilize in the next step.
  - b. The algorithm then finds a transition matrix from  $L_t$  to  $L_k$  that maximizes welfare when the starting distribution on layer  $t$  is  $D_t$  and the remaining transition matrices are fixed as in the solution to the corresponding sub-problem. Although the overall welfare-maximization problem is non-convex, this sub-problem can be solved as a linear program (Program 6) because all transition matrices except for one have been fixed as the solution to our sub-problem.
  - c. Finally, the algorithm picks and stores the recovered transition matrices from layer  $L_t$  to  $L_k$  that yield the highest reward, among all the transition matrices recovered from step 1b.

We remark that while (for notational simplicity) our algorithm is written as if all layers have size exactly  $w$ , it can easily be extended to the case in which all layers have size *at most*  $w$ .

We briefly note why Program (6) is a linear program. The objective is linear because only the matrix  $M_t$  represents variables. Thus we simply need to verify that the constraint on the cost is linear.

► **Definition 7.** We say that a transition matrix  $M_t \in \mathcal{M}$  is feasible with respect to a budget split  $B_{\geq t}, B_{\geq t+1}$  if and only if

$$c(M_t, M_t^0) \leq B_{\geq t+1} - B_{\geq t}.$$

and  $M_t(i, j) = M_t^0(i, j)$  for every non-malleable edge  $(i, j)$ .

Note that saying that  $M_t$  feasible with respect to  $B_{\geq t}, B_{\geq t+1}$  is equivalent to saying that  $M_t$  is a feasible solution to Program (6) with parameters  $B_{\geq t}, B_{\geq t+1}, D_t, D_{t+1}$  for any  $D_t, D_{t+1} \in \mathcal{D}(\varepsilon)$ . The constraint  $c(M_t, M_t^0) \leq B_{\geq t+1} - B_{\geq t}$  can be equivalently replaced by  $2w^2 + 1$  linear constraints. To do so, we introduce  $w^2$  variables -  $a_1, a_2, \dots, a_{w^2}$ . The constraint can then be rewritten in the form  $\sum_{i=1}^{w^2} |f_i| \leq B_{\geq t+1} - B_{\geq t}$ , where each  $f_i$  is a linear combination of the variables. We can thus express the budget constraint of Program 6 by the following set of linear constraints:



■ **Algorithm 1** Dynamic Program for (Approximate) Social Welfare Maximization.

---

**Input:** Input distribution  $D_1$ , reward vector  $R$ , initial transition matrices  $M_1^0, \dots, M_{k-1}^0$ , budget  $B$ , discretization parameter  $\varepsilon$ .

**Output:** Transition  $M(B^\varepsilon, D_1^0)$  from  $L_1$  to  $L_k$ .

**Initialization:** Let  $B_{\geq k} = 0$ ,  $M(B_{\geq k}, D_k) = I$ ,  $B^\varepsilon = \max\{x \in \mathcal{B}(\varepsilon) : x \leq B\}$ .

**for** layer  $t = k - 1, \dots, 1$  **do**

**for** all distributions  $D_t \in \mathcal{D}(\varepsilon)$  if  $t \neq 1$  ( $D_t = D_1^0$  if  $t = 1$ ) and budgets  $B_{\geq t} \in \mathcal{B}(\varepsilon)$  with  $B_{\geq t} \leq B$  **do**

**for** all distributions  $D_{t+1} \in \mathcal{D}(\varepsilon)$  and budgets  $B_{\geq t+1} \leq B_{\geq t}$  such that  $B_{\geq t+1} \in \mathcal{B}(\varepsilon)$  **do**

Solve linear program

$$M_t(B_{\geq t}, B_{\geq t+1}, D_t, D_{t+1}) = \arg \max_{M_t} R^\top M(B_{\geq t+1}, D_{t+1}) M_t D_t$$

s.t.  $c(M_t, M_t^0) \leq B_{\geq t} - B_{\geq t+1}$ ,

$$M_t(i, j) = M_t^0(i, j) \quad \forall (i, j) \in \overline{E_t^{mal}}$$

$$M_t \in \mathcal{M} \tag{6}$$

**end**

Pick  $B_{\geq t+1}, D_{t+1}$  leading to the highest objective value in Program 6, and set  $M(B_{\geq t}, D_t) = M(B_{\geq t+1}, D_{t+1}) M_t(B_{\geq t}, B_{\geq t+1}, D_t, D_{t+1})$ .

**end**

**end**

**Return**  $M(B^\varepsilon, D_1)$ .

---

1.  $f_i \leq a_i \quad \forall i \in [w^2]$
2.  $-f_i \leq a_i \quad \forall i \in [w^2]$
3.  $\sum_{i=1}^{w^2} a_i \leq B_{\geq t+1} - B_{\geq t}$ .

Thus, Program 6 can be written as a linear program with the number of constraints and variables being polynomial in  $w$ .

## 4.2 Running Time and Social Welfare Guarantees

We provide the running time and social welfare guarantees of Algorithm 8 below.

► **Theorem 8.** *Algorithm 1 instantiated with discretization parameter  $\varepsilon$  yields a solution achieving social welfare at least  $OPT - 3(k-1)\varepsilon\|R\|_\infty$ , and has running time  $O\left(k\frac{B}{\varepsilon}\left(\frac{1}{\varepsilon}\right)^{w^2}f(w)\right)$ , where  $f(w)$  is any upper-bound on the running time for solving linear Program 6, which is always polynomial in  $w$ .*

This immediately yields the following corollary:

► **Corollary 9.** *Algorithm 1 with discretization parameter  $\varepsilon' = \frac{\varepsilon}{3(k-1)}$  yields social welfare at least  $OPT - \varepsilon\|R\|_\infty$ , and has running time  $O\left(k^2\frac{B}{\varepsilon}\left(\frac{k}{\varepsilon}\right)^{w^2}f(w)\right)$ , where  $f(w)$  is any upper-bound on the running time for solving linear Program 6, which is always polynomial in  $w$ .*

We observe that this running time is polynomial in  $k$  (the depth of the network) and  $1/\varepsilon$  (the inverse additive error tolerance), but exponential in  $w$  (the width of the network). Hence our algorithm runs in polynomial time for the class of constant width networks.



► **Remark 10.** We note that our additive near-optimality guarantee can be translated into a multiplicative guarantee. In the case where *all edges are malleable*, this follows from noting that given budget  $B$ ,  $OPT \geq \frac{B}{2w} \|R\|_\infty$ : this can be reached by investing the totality of the budget into transitioning every node in the second-to-last layer to the highest reward node in the last layer, with probability  $\frac{B}{2w}$  for each such node. Taking  $\varepsilon = \delta \cdot \frac{B}{6(k-1)w}$  for some constant  $\delta < 1$  gives a multiplicative approximation to the optimal social welfare with approximation factor  $1 - \delta$ .

For the case in which non-malleable edges are allowed, a lower bound on  $OPT$  is given by  $OPT \geq W_0$ . Taking  $\varepsilon = \delta \cdot \frac{W_0}{3(k-1)\|R\|_\infty}$  yields a multiplicative  $1 - \delta$  approximation still.

### Proof of Theorem 8

The proof of Theorem 8 relies on the following lemma, and its corollary:

► **Lemma 11.** *Let  $M \in \mathbb{R}^{w \times w}$  be a left stochastic matrix, and let  $D, D' \in \mathcal{D}$  be probability distributions.*

$$\|MD - MD'\|_1 \leq \|D - D'\|_1.$$

**Proof.** Note that

$$\begin{aligned} \|M(D - D')\|_1 &= \sum_{i=1}^w |(M(D - D'))(i)| = \sum_{i=1}^w \left| \sum_{j=1}^w M(i, j)(D(j) - D'(j)) \right| \\ &\leq \sum_{i=1}^w \sum_{j=1}^w |M(i, j)(D(j) - D'(j))| \\ &= \sum_{j=1}^w |D(j) - D'(j)| \sum_{i=1}^w |M(i, j)| \\ &= \sum_{j=1}^w |D(j) - D'(j)| \\ &= \|D - D'\|_1, \end{aligned}$$

where the inequality follows from the triangle inequality, and the second-to-last equality from the fact that

$$\sum_{i=1}^w |M(i, j)| = \sum_{i=1}^w M(i, j) = 1 \quad \forall j \in [w]$$

as  $M$  is a left stochastic matrix. ◀

► **Corollary 12.** *Let  $R \in \mathbb{R}^w$  be a real vector and  $D, D' \in \mathcal{D}$  be probability distributions such that  $\|D - D'\|_1 \leq \varepsilon$ , and  $M \in \mathbb{R}^{w \times w}$  a left stochastic matrix. Then*

$$R^\top MD \geq R^\top MD' - \|R\|_\infty \cdot \varepsilon.$$

**Proof of Corollary 12.**  $\|R^\top M(D' - D)\|_1 \leq \|R\|_\infty \|M(D' - D)\|_1 \leq \|R\|_\infty \|D' - D\|_1 \leq \|R\|_\infty \cdot \varepsilon$ , where the first step follows from Holder's inequality. ◀

We are now ready to prove Theorem 8:

**Proof of Theorem 8.** Let us denote by  $B_1^\varepsilon, \dots, B_{k-1}^\varepsilon$  a split of the budget for the discretized problem with  $B_{\geq t}^\varepsilon = B_t^\varepsilon + \dots + B_{k-1}^\varepsilon$ . Let  $M_1^\varepsilon, \dots, M_{k-1}^\varepsilon$  a set of transition matrices achieving welfare  $R^\top M_{k-1}^\varepsilon \dots M_1^\varepsilon D_1^0 \geq OPT^\varepsilon \triangleq OPT - (k-1)\varepsilon \|R\|_\infty$  that is feasible with respect to budget split  $B_1^\varepsilon, \dots, B_{k-1}^\varepsilon$ . Note that such a budget split and matrices exist by Claim 4. Let  $D_t^\varepsilon$  the probability distribution on layer  $t$  defined by these transition matrices, i.e.

$$D_t^\varepsilon = M_{t-1}^\varepsilon \dots M_1^\varepsilon D_1^0.$$

To prove the result, we will show by induction that for all  $B_{\geq t} \geq B_{\geq t}^\varepsilon$ , and for  $D_t \in \mathcal{D}(\varepsilon)$  such that  $\|D_t - D_t^\varepsilon\|_1 \leq \varepsilon$ ,

$$R^\top M(B_{\geq t}, D_t) D_t \geq OPT^\varepsilon - 2(k-t)\varepsilon \|R\|_\infty.$$

This will directly imply that as  $B^\varepsilon$  is one of the possible values of  $B_{\geq 1}$ ,

$$R^\top M(B^\varepsilon, D_1) D_1 \geq OPT^\varepsilon - 2(k-1)\varepsilon \|R\|_\infty.$$

Combined with Claim 4 that states  $OPT^\varepsilon \geq OPT - (k-1)\varepsilon \|R\|_\infty$ , we will obtain the result.

Let us now provide our inductive proof. First, consider the transition from layer  $L_{k-1}$  to layer  $L_k$ . Note that

$$OPT^\varepsilon \leq R^\top M_{k-1}^\varepsilon \dots M_1^\varepsilon D_1^0 = R^\top M_{k-1}^\varepsilon D_{k-1}^\varepsilon.$$

Let  $D_{k-1} \in \mathcal{D}(\varepsilon)$  be such that  $\|D_{k-1} - D_{k-1}^\varepsilon\|_1 \leq \varepsilon$ . Note then that by Corollary 12,

$$R^\top M_{k-1}^\varepsilon D_{k-1} \geq R^\top M_{k-1}^\varepsilon D_{k-1}^\varepsilon - \varepsilon \|R\|_\infty.$$

Further,  $M_{k-1}^\varepsilon$  is feasible for Program (6) with respect to  $B_{\geq k-1}, B_{\geq k} = 0$ , given  $B_{\geq k-1} \geq B_{\geq k-1}^\varepsilon$ . As such, for  $B_{\geq k-1} \geq B_{\geq k-1}^\varepsilon$ , we have that

$$R^\top M(B_{\geq k-1}, D_{k-1}) D_{k-1} \geq R^\top M_{k-1}^\varepsilon D_{k-1},$$

and in turn

$$R^\top M(B_{\geq k-1}, D_{k-1}) D_{k-1} \geq OPT^\varepsilon - \varepsilon \|R\|_\infty.$$

Now, suppose the induction hypothesis holds at layer  $t+1$ . I.e., for all  $B_{\geq t+1} \geq B_{\geq t+1}^\varepsilon$ , for  $D_{t+1} \in \mathcal{D}(\varepsilon)$  such that  $\|D_{t+1} - D_{t+1}^\varepsilon\|_1 \leq \varepsilon$ ,

$$R^\top M(B_{\geq t+1}, D_{t+1}) D_{t+1} \geq OPT^\varepsilon - 2(k-t-1)\varepsilon \|R\|_\infty.$$

For any  $B_{\geq t} \geq B_{\geq t}^\varepsilon$ , note that one can set  $B_{\geq t+1} = B_{\geq t+1}^\varepsilon$  and  $B_t \geq B_t^\varepsilon$ ; hence,  $M_t^\varepsilon$  is feasible for Program (6) with respect to  $B_t \geq B_t^\varepsilon, B_{\geq t+1}^\varepsilon$ . Since  $\|D_t - D_t^\varepsilon\|_1 \leq \varepsilon$  and  $\|D_{t+1} - M_t^\varepsilon D_t^\varepsilon\|_1 \leq \varepsilon$ , we have that by Corollary 12,

$$\begin{aligned} R^\top M(B_{\geq t+1}, D_{t+1}) M_t^\varepsilon D_t &\geq R^\top M(B_{\geq t+1}, D_{t+1}) M_t^\varepsilon D_t^\varepsilon - \varepsilon \|R\|_\infty \\ &\geq R^\top M(B_{\geq t+1}, D_{t+1}) D_{t+1} - 2\varepsilon \|R\|_\infty. \end{aligned}$$

Using the induction hypothesis, we obtain that  $R^\top M(B_{\geq t+1}, D_{t+1}) M_t^\varepsilon D_t \geq OPT^\varepsilon - 2(k-t)\varepsilon \|R\|_\infty$ . In particular, we get

$$R^\top M(B_{\geq t}, D_t) D_t \geq OPT^\varepsilon - 2(k-t)\varepsilon \|R\|_\infty,$$

which concludes the proof of the social welfare guarantee. For the running time, we note that at each time step  $t$ , we solve one instance of Program 6 for each of the (at most)  $\frac{B}{\varepsilon}$  possible budget splits of  $B_{\geq t}$  and for each of the  $\left(\frac{1}{\varepsilon}\right)^w$  (by Claim 6) probability distributions in  $\mathcal{D}(\varepsilon)$  in layer  $L_t$  and layer  $L_{t+1}$ ; i.e., for each  $t$ , the algorithm solves  $O\left(\frac{B}{\varepsilon} \left(\frac{1}{\varepsilon}\right)^{w^2}\right)$  optimization programs. Then, the algorithm finds the solution of all of these programs with the best objective value, which can be done in time linear in the number of such solutions, i.e.  $O\left(\frac{B}{\varepsilon} \left(\frac{1}{\varepsilon}\right)^{w^2}\right)$ . This is repeated for  $k-1$  values of  $t$ .  $\blacktriangleleft$

## 5 (Ex-post) Maximin Value Maximization

Although social welfare maximization is a natural objective, it is well-known that it can be “unfair” in the sense that it explicitly prioritizes the welfare of larger populations (here represented as initial positions that have larger probability mass) over smaller populations. We can alternately evaluate a solution according to the welfare of the *least-well-off* population (here represented by the initial position with the smallest expected value) and ask to optimize *that* objective. We show how to optimize this objective in this section, when one demands a deterministic solution.

### 5.1 A Dynamic Programming Algorithm for Computing an Ex-post Maximin Allocation

In this subsection, we adapt the dynamic programming approach in Section 4.1 to give an approximation algorithm for the problem of maximizing the minimum expected reward over all initial positions. Recall that  $\mathcal{D}$ , the probability simplex, denotes the set of all possible probability distributions on a layer. Intuitively, our algorithm for maximizing social welfare kept track of a single probability distribution in each subproblem: the overall probability of arriving at each vertex in the layer *over both the randomness of an individual’s initial position, and the randomness of the transition matrix*. In order to optimize the *minimum* expected value over all initial positions, we will need to keep track of more state. At every layer  $L_t$ , we will keep track of the probability of reaching each vertex in that layer *from each initial position in the starting layer*. So, we will now keep track of collections of  $w$  probability distributions in  $\mathcal{D}^w$ , one for each starting position. We call the elements of  $\mathcal{D}^w$  *population-wise distributions*.

We introduce a discretization  $\mathcal{A}(\varepsilon)$  of  $\mathcal{D}^w$ , as follows:  $\mathcal{A}(\varepsilon) \triangleq (\mathcal{D}(\varepsilon))^w$ , where  $\mathcal{D}(\varepsilon)$  denotes a  $\varepsilon$ -net of  $\mathcal{D}$  (of size  $(\frac{1}{\varepsilon})^w$ ). Given a population-wise probability distribution  $A_t \in \mathcal{A}(\varepsilon)$  at layer  $t$ , we write  $A_t^j$  for the probability distribution corresponding to population  $j$ . The algorithm works as follows, just as before, running backwards from the final layer to the first layer. We describe the algorithm below informally, a formal presentation may be found in the full version [1].

#### 5.1.1 Algorithm

1. For  $t$  going backwards from  $k - 1$  to 1, the algorithm does the following, for every population-wise distribution  $A_t \in \mathcal{A}(\varepsilon)$  on  $L_t$ :
  - a. The algorithm explores all splits of the budget  $B_{\geq t}$  for layers  $t$  to  $k$  into a budget  $B_t$  for the transition from  $L_t$  to  $L_{t+1}$  and a budget  $B_{\geq t+1}$  for  $L_{t+1}$  to  $L_k$ , as well as all choices of output population-wise probability distributions  $A_{t+1} \in \mathcal{A}(\varepsilon)$  on layer  $L_{t+1}$ .
  - b. The algorithm then finds a near-optimal transition matrix from  $L_t$  to  $L_k$  for every budget decomposition, by using the previously computed near-optimal solution for layers  $L_{t+1}$  to  $L_k$ , and solving a program similar to Program 6. The program maximizes the *minimum reward obtained from any initial position*, assuming the population-wise distribution of individuals at layer  $L_t$  is given by  $A_t$ .
  - c. Finally, the algorithm picks and stores the best recovered transition matrices from layer  $L_t$  to  $L_k$  that yield the highest reward, among all the transition matrices recovered from step 1b.

The input population-wise probability distribution  $A_1 \in \mathcal{D}^w$  on the first layer is defined in the following manner,  $A_1^j := e_j$  (the  $j$ -th basis vector in the usual orthonormal basis of  $\mathbb{R}^w$ ) for all  $j \in [w]$ .

Note that the Program in step 1b can be written as a linear program of size polynomial in  $w$ , using the same method that was employed to write Program (6) as a linear program.

## 5.2 Running Time and Ex-Post Maximin Value Guarantees

Remember that we let  $OPT_{MM}$  denote the maximin value of the given network. The running time and accuracy guarantees of the described Algorithm are provided below:

► **Theorem 13.** *The Algorithm described in Section 5.1.1 with discretization parameter  $\varepsilon$  yields maximin value at least  $OPT_{MM} - 3(k-1)\varepsilon\|R\|_\infty$ , and has running time  $O\left(k\frac{B}{\varepsilon}\left(\frac{1}{\varepsilon}\right)^{w^4}g(w)\right)$ , where  $g(w)$  is any upper-bound on the running time for solving the linear Program in Step 1b, which is always polynomial in  $w$ .*

This immediately implies the following corollary:

► **Corollary 14.** *The algorithm described in 5.1.1 with discretization parameter  $\varepsilon' = \frac{\varepsilon}{3(k-1)}$  yields maximin value at least  $OPT_{MM} - \varepsilon\|R\|_\infty$ , and has running time  $O\left(k^2\frac{B}{\varepsilon}\left(\frac{k}{\varepsilon}\right)^{w^4}g(w)\right)$ , where  $g(w)$  is a polynomial upper-bound on the running time of the linear Program in Step 1b.*

The proof of Theorem 13 is almost identical to that of Theorem 8. We provide a complete proof in the full version [1].

## 6 (Ex-ante) Maximin Value Maximization

In this section, we consider the problem of optimizing the *ex-ante* minimum expected value over all initial positions: in other words, we allow ourselves to find a *distribution* over solutions, and take expectations over the randomness of this distribution, solving:

$$\begin{aligned} \max_{\Delta M} \quad & \min_{j \in [w]} R^\top \mathbb{E}_{M \sim \Delta M} [M_{k-1} \dots M_1] e_j \\ \text{s.t.} \quad & \Delta M \in \Delta \mathcal{F}(B, M_1^0, \dots, M_{k-1}^0) \end{aligned} \quad (7)$$

We show in the full version [1] that this can yield strictly higher utility than optimizing the ex-post minimum value. We then give an algorithm for solving the ex-ante problem by exhibiting a game theoretic reduction to the ex-post problem.

### 6.1 Solving the Ex-ante Maximization Problem Using Algorithm 1

Because Program 3 is a max min problem over a polytope, we can view it as a zero-sum game, and the solution that we want corresponds to a maximin equilibrium strategy of this game. As first shown by Freund and Schapire[11], it is possible to compute an approximate equilibrium of a zero-sum game if we can implement a *no-regret* learning algorithm for one of the players, and an approximate best-response algorithm for the other player: if we simply simulate repeated play of the game between a no-regret player and a best-response player, then the empirical average of player actions in this simulation converges to the Nash equilibrium of the game.

This forms the basis of our algorithm. One player plays the “multiplicative weights” algorithm over the initial positions in layer 1 of the graph. This induces at every round a distribution over initial positions. The best response problem, which must be solved by the other player, corresponds to solving a welfare-maximization problem given the distribution over initial positions represented by the multiplicative weights distribution. Fortunately, this

is exactly the problem that we have already given a dynamic programming solution for. The solution in the end corresponds to the uniform distribution over the solutions computed by the best-response player over the course of the dynamics. The algorithm is described formally in the full version [1].

■ **Algorithm 2** 2-Player Dynamics for the Ex-Ante Maximin Problem.

---

**Input:** Time horizon  $T$ , reward vector  $R$  on layer  $L_k$ , initial transition matrices  $M_1^0, \dots, M_{k-1}^0$ , budget  $B$ , discretization parameter  $\varepsilon$ .

**Output:**  $M^1, \dots, M^T \in \mathcal{F}(B, M_1^0, \dots, M_{k-1}^0)$ .

**Initialization:** The no-regret player picks  $D^1 = (\frac{1}{w}, \dots, \frac{1}{w}) \in \mathcal{D}$ , the uniform distribution over  $[w]$ .

**for**  $t = 1, \dots, T$  **do**

The no-regret player plays distribution  $D^t \in \mathcal{D}$ .

The best-response player chooses  $M^t \in \mathcal{F}(B, M_1^0, \dots, M_{k-1}^0)$  such that

$$R^\top M_{k-1}^t \dots M_1^t D^t \geq \max_{M \in \mathcal{F}} R^\top M_{k-1} \dots M_1 D^t - \varepsilon \|R\|_\infty,$$

using Algorithm 1.

The no-regret player observes  $u_i^t = \frac{R^\top M_{k-1}^t \dots M_1^t e_i}{\|R\|_\infty}$  for all  $i \in [w]$ , and picks  $D^{t+1}$  via multiplicative weight update, as follows:

$$D^{t+1}(i) = \frac{D^t(i) \beta^{u_i^t}}{\sum_{j=1}^w D^t(j) \beta^{u_j^t}} \quad \forall i \in [w],$$

with  $\beta = \frac{1}{1 + \sqrt{2 \frac{\ln w}{T}}} \in [0, 1)$ .

**end**

---

► **Lemma 15.** *Let  $T > 0$ ,  $\overline{\Delta M}$  be the probability distribution that picks  $(M_1, \dots, M_{k-1}) \in \mathcal{F}(B, M_1^0, \dots, M_{k-1}^0)$  with probability*

$$\frac{1}{T} \sum_{t=1}^T \mathbb{1} \{ (M_1, \dots, M_{k-1}) = (M_1^t, \dots, M_{k-1}^t) \},$$

*where  $M^1, \dots, M^T$  are the outputs of Algorithm 2. Then  $\overline{\Delta M} \left( \varepsilon + \sqrt{2 \frac{\ln w}{T}} + \frac{\ln w}{T} \right) \|R\|_\infty$ -approximately optimizes Program 3.*

The proof of Lemma 15 follows from interpreting Program 3 as zero-sum game, noting that the best response problem for the maximization player corresponds to the welfare-maximization problem for which we have an efficient algorithm, and then applying the no-regret dynamic analysis from Freund and Schapire [11]. The details are provided in the full version [1].

## 7 Price of Fairness

In this section, we compute lower bounds on a notion of “price of fairness”, and we show these lower bounds are tight when restricting attention to pipelines whose edges are *all* malleable. Specifically, we compare the optimal welfare achievable with the welfare that is

achievable if we instead use our budget to maximize the *minimum* value over initial positions – i.e. if we solve the maximin problem. We focus on the ex-post maximin problem – i.e. we prove our bounds with respect to *deterministic* solutions. We note that there may be many different maximin optimal solutions that differ in their overall welfare, and so we consider two variants of the price of fairness in our setting – comparing with both the *maximum* welfare consistent with a maximin optimal solution, and the *minimum* welfare consistent with a maximin optimal solution.

Let  $OPT_{SW}$  the optimal value of Program (1) (the optimal social welfare). Let  $S^f$  be the set of solutions to Program (2) (the deterministic maximin problem). Further, define

$$W(M_1, \dots, M_{k-1}) \triangleq R^\top M_{k-1} \dots M_1 D_1^0$$

to be the social welfare achieved by transition matrices  $M_1, \dots, M_{k-1}$ , and

$$W_{fair}^+ \triangleq \max_{(M_1, \dots, M_{k-1}) \in S^f} W(M_1, \dots, M_{k-1}),$$

$$W_{fair}^- \triangleq \min_{(M_1, \dots, M_{k-1}) \in S^f} W(M_1, \dots, M_{k-1})$$

to be the maximum and minimum social welfare respectively that are consistent with maximin optimal solutions. We define two variants of “the price of fairness” in our setting as:  $P_f^+ \triangleq \frac{OPT_{SW}}{W_{fair}^+} \geq 1$ , and

$$P_f^- \triangleq \frac{OPT_{SW}}{W_{fair}^-} \geq 1.$$

Note that  $P_f^+ \leq P_f^-$  always, as  $P_f^+$  compares the optimal social welfare with the solution of Program 2 with highest social welfare, while  $P_f^-$  considers the solution that has the lowest social welfare. We provide matching lower bounds on  $P_f^+$  and upper bounds on  $P_f^-$ . This, in turn, provides tight bounds on the price of fairness with respect to *any* choice of maximin solution.

## 7.1 Lower Bounds on $P_f^+$

Our lower bounds are based on the following construction:

► **Example 16.** Consider a network with only two layers,  $L_1$  and  $L_2$ , such that  $L_1$  has  $w$  nodes and  $L_2$  has 2 nodes. Suppose the starting distribution is given by  $D_1^0 = (1 - (w-1)\varepsilon, \varepsilon, \dots, \varepsilon)^\top$  for  $\varepsilon > 0$  small enough, the reward vector is given by  $R = (1, 0)^\top$ , and the initial transition matrix  $M_1^0$  is given by

$$M_1^0 = \begin{pmatrix} 0 & \dots & 0 \\ 1 & \dots & 1 \end{pmatrix}.$$

I.e., in the initial transition matrix, every starting node transitions to the destination node that has reward 0, and the welfare of the initial network is 0. We assume all edges are malleable.

► **Theorem 17.** *For all  $w \in \mathbb{N}$ , for any  $\delta > 0$ , there exists a network with  $k = 2$  with price of fairness*

$$P_f \geq \begin{cases} w - \delta & \text{if } 0 < B \leq 2 \\ \frac{2w}{B} - \delta & \text{if } 2 < B \leq 2w \\ 1 & \text{if } B \geq 2w \end{cases}.$$

The proof follows from solving the social welfare maximization problem and the maximin value problem on Example 16. The complete proof is provided in the full version [1].

## 7.2 Upper Bounds on $P_f^-$

Importantly, in this section, we restrict ourselves to pipelines such that *all* edges are malleable. In this case, we show upper bounds that tightly match the lower bounds of Section 7.1.

Our upper bounds will make use of the following claim, which bounds the maximum social welfare that can be achieved under budget  $B$ .

► **Lemma 18.**

$$OPT_{SW} \leq \|R\|_\infty \text{ and } OPT_{SW} \leq W^0 + \frac{B}{2} \|R\|_\infty,$$

where  $W^0 = R^\top M_{k-1}^0 \dots M_1^0 D_1^0$  is the initial welfare.

The proof of this lemma is straightforward and is deferred to the full version [1]. We will also need lower bounds on the social welfare achieved by any optimal solution to the maximin program. The first lower bound is a function of  $B$  and  $w$ , but is independent of  $W^0$ .

► **Lemma 19.** *When all edges are malleable, for any  $(M_1^f, \dots, M_{k-1}^f) \in S^f$ ,*

$$W(M_1^f, \dots, M_{k-1}^f) \geq \min\left(1, \frac{B}{2w}\right) \|R\|_\infty.$$

The proof of Lemma 19 is deferred to the full version [1]. The second lower bound we need shows that the social welfare achieved by a solution to Program (2) is lower-bounded by the initial social welfare  $W^0 = R^\top M_{k-1}^0 \dots M_1^0 D_1^0$ .

► **Lemma 20.** *When all edges are malleable, for any  $(M_1^f, \dots, M_{k-1}^f) \in S^f$ ,*

$$W(M_1^f, \dots, M_{k-1}^f) \geq W^0.$$

We defer the full proof of Lemma 20 to the full version [1]. We can now use Lemmas 18, 19 and 20 to derive nearly tight upper bounds on the price of fairness with respect to the worst maximin solution:

► **Theorem 21.** *For every instance of the problem in which edges are malleable, we have that*

$$P_f^- \leq \begin{cases} w + 1 & \text{if } 0 < B \leq 2 \\ \frac{2w}{B} & \text{if } 2 < B \leq 2w \\ 1 & \text{if } B \geq 2w \end{cases}.$$

**Proof.** We divide the proof in three cases:

1.  $B \geq 2w$ . By Lemma 19, it must be the case that any optimal solution to Program (2) has welfare at least  $\min(1, \frac{B}{2w}) \|R\|_\infty = \|R\|_\infty$ . It is then immediately the case that  $OPT_{SW} = \|R\|_\infty$  by Lemma 18 and  $P_f^- = 1$ .
2.  $2 < B \leq 2w$ . By Lemma 18, we have  $OPT_{SW} \leq \|R\|_\infty$ . Further, by Lemma 19, we have that any solution to Program (2) has welfare at least  $\frac{B}{2w} \|R\|_\infty$ . This immediately yields the result.

3.  $0 < B \leq 2$ . By Lemma 18, we have  $OPT_{SW} \leq W^0 + \frac{B}{2} \|R\|_\infty$ . By Lemmas 19 and 20, we have that the social welfare of any maximin solution is at least  $W^0$  and at least  $\frac{B}{2w} \|R\|_\infty$ . Therefore, the price of fairness is upper-bounded on the one hand by

$$P_f^- \leq \frac{W^0 + \frac{B}{2} \|R\|_\infty}{W^0} = 1 + \frac{\frac{B}{2} \|R\|_\infty}{W^0}$$

and on the other hand by

$$P_f^- \leq \frac{W^0 + \frac{B}{2} \|R\|_\infty}{\frac{B}{2w} \|R\|_\infty} = w + \frac{W^0}{\frac{B}{2w} \|R\|_\infty}.$$

When  $W^0 \geq \frac{B}{2w} \|R\|_\infty$ , the first bound gives  $P_f^- \leq 1 + \frac{\frac{B}{2} \|R\|_\infty}{\frac{B}{2w} \|R\|_\infty} = w + 1$ ,

and when  $W^0 \leq \frac{B}{2w} \|R\|_\infty$ , the second bound yields  $P_f^- \leq w + \frac{\frac{B}{2w} \|R\|_\infty}{\frac{B}{2w} \|R\|_\infty} = w + 1$ , which concludes the proof.  $\blacktriangleleft$

## 8 Hardness of Approximation

In this section, we show that the problem of finding the ex-post maximin value of a pipeline intervention problem instance within an approximation factor of 2 is NP-hard in the general case, where the width  $w$  of the network is not bounded. More specifically, we show that no algorithm that has a time bound polynomial in  $w, k$  and  $B$  can give a 2-approximation to the maximin value unless  $P = NP$ . This hardness result holds for  $k$  as small as 17. We remark that our result and proof can be immediately extended to show hardness of  $C$ -approximation, for any constant  $C$ , for an appropriate choice of constant depth  $k$ .

We show this hardness result via a reduction from a gap version of the vertex cover problem. The result of Dinur and Safra [8] shows that it is NP-hard to approximate the minimum vertex cover by a factor smaller than 1.306. In particular, their result shows that the following gap version of vertex cover is NP-hard: given  $(\mathcal{G}, \kappa)$ , we wish to either know if the graph  $\mathcal{G}$  has a vertex cover of size  $\kappa$ , or has no vertex cover smaller than size  $1.306\kappa$ .

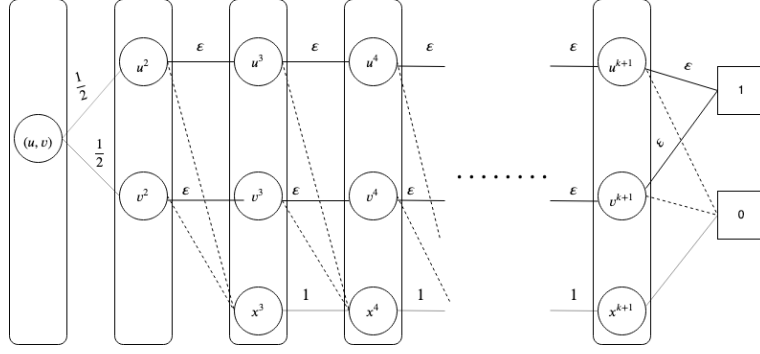
We provide a description of our reduction below, and defer a formal statement along with a complete proof to the full version [1].

### 8.1 The Reduction

Our reduction works as follows: we construct a pipeline intervention instance of constant width (17 layers) from the given graph. The first layer has a node corresponding to each edge  $(u, v)$  of the original graph, and is connected by edges to nodes corresponding to vertices  $u$  and  $v$  on the second layer. We set up the instance so that positive probability mass is only ever added to a set of edge disjoint paths, where each path corresponds to a vertex in the original graph. These paths are shown by the dark, solid lines in Figure 1. The main idea behind the reduction is the following - by observing how allocations finding the maximin value split the budget over these edge disjoint paths, we can find out which vertices would form a small vertex cover of the original graph.

Formally, let the given graph  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$  we reduce from have  $n$  vertices ( $|\mathcal{V}| = n$ ) and  $m$  edges ( $|\mathcal{E}| = m$ ). We construct a pipeline intervention problem instance  $\mathcal{I}'$  with  $k+2$  layers and width  $w$ , where  $k = 15$  and  $w$  is polynomial in  $n$ . The instance  $\mathcal{I}'$  has an associated budget  $B(\kappa, \varepsilon) = 2k\kappa\varepsilon$  where  $\varepsilon < \frac{1}{2}$ . For the sake of clarity, we refer to the set of vertices  $\mathcal{V}$  in the vertex cover instance as “vertices” and the vertices in the instance  $\mathcal{I}'$  as “nodes”. A complete description of instance  $\mathcal{I}'$  is as follows:





■ **Figure 1** Constructed Instance of the Pipeline Intervention problem.

1. The first layer,  $L_1$ , has exactly  $m$  nodes, with each edge  $(u, v)$  in graph  $\mathcal{G}$  having a unique corresponding node of the same label in layer  $L_1$ .
2. The second layer has exactly  $n$  nodes, with each vertex  $v$  in graph  $\mathcal{G}$ , having a unique corresponding node in layer  $L_2$  with label  $v^2$ .
3. The next  $k - 1$  layers are of the following form - layer  $L_i$ , for  $i = 3$  to  $k + 1$ , has  $n + 1$  nodes. The first  $n$  nodes have labels from the set  $\{v^i\}_{v \in V}$ , i.e., each vertex  $v$  in the original graph  $\mathcal{G}$  has a corresponding node  $v^i$  in layer  $L_i$ . The last node is indexed by  $x^i$  and exists to capture the “leftover” outward probability from the nodes  $\{v^{i-1}\}_{v \in V}$  in layer  $L_{i-1}$ .
4. The final layer  $L_{k+2}$  has two reward nodes -  $y$ , of reward 1 and  $z$ , of reward 0.

We now describe the initial transition matrices.

1. From layer  $L_1$  to layer  $L_2$ : for every node  $(u, v)$  in layer  $L_1$ , the outgoing probability is equally split between edges to nodes  $u^1$  and  $v^1$  in layer  $L_2$ , i.e., edges  $((u, v), u^1)$  and  $((u, v), v^1)$  each have probability  $\frac{1}{2}$ .
2. From layer  $L_i$  to layer  $L_{i+1}$  for  $i = 2$  to  $k$ : For all vertices  $v \in \mathcal{V}$  (i.e., the original graph), the corresponding edge  $(v^i, v^{i+1})$  (in our construction) has probability  $\epsilon$ . The remaining outgoing probability out of node  $v^i$  goes to the leakage node  $x^{i+1}$ . We call edges of the form  $(v^i, x^{i+1})$  “leakage” edges. For  $i \geq 3$ , the edge  $(x^i, x^{i+1})$  has all the outward probability, i.e., 1, from node  $x^i$ .
3. From layer  $L_{k+1}$  to layer  $L_{k+2}$ : each node in layer  $L_{k+1}$  is connected to  $z$ , the zero reward node, with probability 1.

We let  $P_v$  be the path going through nodes  $v^2, v^3 \dots v^{k+1}, y$  in our construction. We will refer to  $\{P_v\}_{v \in \mathcal{V}}$  as vertex paths. Let  $E'$  be the set of edges found on paths  $\{P_v\}_{v \in \mathcal{V}}$ . Let  $E''$  contain all the “leakage” edges in the instance  $\mathcal{I}'$ , i.e., edges of the form  $(v^i, x^{i+1})$  as well as all edges of the form  $(v^{k+1}, z)$ . We stipulate, as part of the description of the instance, that  $E' \cup E''$  is the set of malleable edges in  $\mathcal{I}'$  and that the probability mass on any other edge cannot be changed. This completes the description of instance  $\mathcal{I}'$ .

## References

- 1 Eshwar Ram Arunachaleswaran, Sampath Kannan, Aaron Roth, and Juba Ziani. Pipeline interventions, 2020. [arXiv:2002.06592](#).
- 2 Siddharth Barman and Sanath Kumar Krishnamurthy. Approximation algorithms for maximin fair division. In *Proceedings of the 2017 ACM Conference on Economics and Computation*, pages 647–664, 2017.

- 3 Matt Barnum. *A new study questions whether Head Start still produces long-run gains seen in past research*, 2019. URL: <https://www.chalkbeat.org/2019/8/8/21108602/a-new-study-questions-whether-head-start-still-produces-long-run-gains-seen-in-past-research>.
- 4 Amanda Bower, Sarah N Kitchen, Laura Niss, Martin J Strauss, Alexander Vargas, and Suresh Venkatasubramanian. Fair pipelines. *arXiv preprint*, 2017. [arXiv:1707.00391](https://arxiv.org/abs/1707.00391).
- 5 Eric Budish. The combinatorial assignment problem: Approximate competitive equilibrium from equal incomes. *Journal of Political Economy*, 119(6):1061–1103, 2011.
- 6 Alexandra Chouldechova and Aaron Roth. The frontiers of fairness in machine learning. *arXiv preprint*, 2018. [arXiv:1810.08810](https://arxiv.org/abs/1810.08810).
- 7 Stephen Coate and Glenn C Loury. Will affirmative-action policies eliminate negative stereotypes? *The American Economic Review*, pages 1220–1240, 1993.
- 8 Irit Dinur and Samuel Safra. On the hardness of approximating minimum vertex cover. *Annals of mathematics*, pages 439–485, 2005.
- 9 Cynthia Dwork and Christina Ilvento. Fairness under composition. In *10th Innovations in Theoretical Computer Science Conference (ITCS 2019)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.
- 10 Dean P Foster and Rakesh V Vohra. An economic argument for affirmative action. *Rationality and Society*, 4(2):176–188, 1992.
- 11 Yoav Freund and Robert E Schapire. Game theory, on-line prediction and boosting. In *Proceedings of the ninth annual conference on Computational learning theory*, pages 325–332, 1996.
- 12 Lily Hu and Yiling Chen. A short-term intervention for long-term fairness in the labor market. In *Proceedings of the 2018 World Wide Web Conference*, pages 1389–1398, 2018.
- 13 Christopher Jung, Sampath Kannan, Changwa Lee, Mallesh M. Pai, Aaron Roth, and Rakesh Vohra. Fair prediction with endogenous behavior. *Manuscript*, 2020.
- 14 Sampath Kannan, Aaron Roth, and Juba Ziani. Downstream effects of affirmative action. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 240–248, 2019.
- 15 Lydia T Liu, Sarah Dean, Esther Rolf, Max Simchowitz, and Moritz Hardt. Delayed impact of fair machine learning. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 6196–6200. AAAI Press, 2019.
- 16 Lydia T Liu, Ashia Wilson, Nika Haghtalab, Adam Tauman Kalai, Christian Borgs, and Jennifer Chayes. The disparate equilibria of algorithmic decision making when individuals invest rationally. *arXiv preprint*, 2019. [arXiv:1910.04123](https://arxiv.org/abs/1910.04123).
- 17 Hussein Mouzannar, Mesrob I Ohannessian, and Nathan Srebro. From fair decision making to social equality. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 359–368, 2019.
- 18 Niche. *Best Public Elementary Schools in America*, 2020. URL: <https://www.niche.com/k12/search/best-public-elementary-schools/>.
- 19 Ariel D Procaccia and Junxing Wang. Fair enough: Guaranteeing approximate maximin shares. In *Proceedings of the fifteenth ACM conference on Economics and computation*, pages 675–692, 2014.
- 20 U.S. News. *Best High Schools Rankings*, 2020. URL: <https://www.usnews.com/education/best-high-schools>.
- 21 Samuel F Way, Daniel B Larremore, and Aaron Clauset. Gender, productivity, and prestige in computer science faculty hiring networks. In *Proceedings of the 25th International Conference on World Wide Web*, pages 1169–1179, 2016.

# A Polynomial Degree Bound on Equations for Non-Rigid Matrices and Small Linear Circuits

Mrinal Kumar

Department of Computer Science and Engineering, IIT Bombay, India  
mrinalkumar08@gmail.com

Ben Lee Volk<sup>1</sup>

Department of Computer Science, University of Texas at Austin, TX, USA  
benleevolk@gmail.com

---

## Abstract

We show that there is an equation of degree at most  $\text{poly}(n)$  for the (Zariski closure of the) set of the non-rigid matrices: that is, we show that for every large enough field  $\mathbb{F}$ , there is a non-zero  $n^2$ -variate polynomial  $P \in \mathbb{F}[x_{1,1}, \dots, x_{n,n}]$  of degree at most  $\text{poly}(n)$  such that every matrix  $M$  which can be written as a sum of a matrix of rank at most  $n/100$  and a matrix of sparsity at most  $n^2/100$  satisfies  $P(M) = 0$ . This confirms a conjecture of Gesmundo, Hauenstein, Ikenmeyer and Landsberg [8] and improves the best upper bound known for this problem down from  $\exp(n^2)$  [11, 8] to  $\text{poly}(n)$ .

We also show a similar polynomial degree bound for the (Zariski closure of the) set of all matrices  $M$  such that the linear transformation represented by  $M$  can be computed by an algebraic circuit with at most  $n^2/200$  edges (without any restriction on the depth). As far as we are aware, no such bound was known prior to this work when the depth of the circuits is unbounded.

Our methods are elementary and short and rely on a polynomial map of Shpilka and Volkovich [17] to construct low degree “universal” maps for non-rigid matrices and small linear circuits. Combining this construction with a simple dimension counting argument to show that any such polynomial map has a low degree annihilating polynomial completes the proof.

As a corollary, we show that any derandomization of the polynomial identity testing problem will imply new circuit lower bounds. A similar (but incomparable) theorem was proved by Kabanets and Impagliazzo [10].

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Algebraic complexity theory; Theory of computation  $\rightarrow$  Circuit complexity

**Keywords and phrases** Rigid Matrices, Linear Circuits, Degree Bounds, Circuit Lower Bounds

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.9

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/2003.12938>.

## 1 Introduction

### 1.1 Equations for varieties in algebraic complexity theory

Let  $V \subseteq \mathbb{F}^n$  be a (not necessarily irreducible) affine variety and let  $\mathbf{I}(V)$  denote its ideal.<sup>2</sup> A non-zero polynomial  $P \in \mathbf{I}(V)$  is called an *equation* for  $V$ . An equation for  $V$  may serve as a “proof” that a point  $\mathbf{x} \in \mathbb{F}^n$  is *not* in  $V$ , by showing that  $P(\mathbf{x}) \neq 0$ .

A fundamental observation of the Geometric Complexity Theory program is that many important circuit lower bounds problems in algebraic complexity theory fit naturally into the setting of showing that a point  $\mathbf{x}$  lies outside a variety  $V$  [15, 5]. In this formulation, one considers  $V$  to be the closure of a class of polynomials of low complexity, and  $\mathbf{x}$  is the coefficient vector of the candidate hard polynomial.

---

<sup>1</sup> A part of this work was done while at the Center for the Mathematics of Information, California Institute of Technology, USA.

<sup>2</sup> For completeness, we provide the formal (standard) definitions for these notions in Subsection 1.4.



Let  $\Delta(V) := \min_{0 \neq P \in \mathcal{I}(V)} \{\deg(P)\}$ . The quantity  $\Delta(V)$  can be thought of as a measure of complexity for the geometry of the variety  $V$ . The quantity  $\Delta(V)$  is a very coarse complexity measure. A recent line of work regarding *algebraic natural proofs* [7, 9] suggests to study the arithmetic circuit complexity of equations for varieties  $V$  that correspond to polynomials with small circuit complexity. Having  $\Delta(V)$  growing like a polynomial in  $n$  is a necessary (but not a sufficient) condition for a variety  $V$  to have an algebraic natural proof for non-containment.

## 1.2 Rigid matrices

A matrix  $M$  is  $(r, s)$ -rigid if  $M$  cannot be written as a sum  $R + S$  where  $\text{rank}(R) \leq r$  and  $S$  contains at most  $s$  non-zero entries. Valiant [18] proved that if  $A$  is  $(\varepsilon n, n^{1+\delta})$ -rigid for some constants  $\varepsilon, \delta > 0$  then  $A$  cannot be computed by arithmetic circuits of size  $O(n)$  and depth  $O(\log n)$ , and posed the problem of *explicitly* constructing rigid matrices with these parameters, which is still open. It is easy to prove that most matrices have much stronger rigidity parameters: over algebraically closed fields a generic matrix is  $(r, (n-r)^2)$ -rigid for any target rank  $r$ .

Let  $\mathbb{F}$  be an algebraically closed field. Let  $A_{r,s} \subseteq \mathbb{F}^{n \times n}$  denote the set of matrices which are not  $(r, s)$ -rigid. Let  $V_{r,s} = \overline{A_{r,s}}$  denote the Zariski closure of  $A_{r,s}$ . A geometric study of  $V_{r,s}$  was initiated by Kumar, Lokam, Patankar and Sarma [11]. Among other results, they prove that for every  $s < (n-r)^2$ ,  $\Delta(V_{r,s}) \leq n^{4n^2}$ . A slightly improved (but still exponential) upper bound was obtained by Gesmundo, Hauenstein, Ikenmeyer and Landsberg [8], who also conjectured that for some  $\varepsilon, \delta > 0$ ,  $\Delta(V_{\varepsilon n, n^{1+\delta}})$  grows like a polynomial function in  $n$ . The following theorem which we prove in this note confirms this conjecture.

► **Theorem 1.** *Let  $\varepsilon < 1/25$ , and let  $\mathbb{F}$  be a field of size at least  $n^2$ . For every large enough  $n$ , there exists a non-zero polynomial  $Q \in \mathbb{F}[x_{1,1}, \dots, x_{n,n}]$ , of degree at most  $n^3$ , which is a non-trivial equation for matrices which are not  $(\varepsilon n, \varepsilon n^2)$ -rigid. That is, for every such matrix  $M$ ,  $Q(M) = 0$ .*

In fact, the conjecture of [8] was slightly weaker: they conjectured that  $\Delta(U)$  is polynomial in  $n$  for every irreducible component  $U$  of  $V_{\varepsilon n, n^{1+\delta}}$ . As shown by [11], the irreducible components are in one-to-one correspondence with subsets of  $[n] \times [n]$  of size  $n^{1+\delta}$  corresponding to possible supports of the sparse matrix  $S$ .

As we observe in Remark 6, it is somewhat simpler to show that each of these irreducible components has an equation with a polynomial degree bound. However, since the number of such irreducible components is exponentially large, it is not clear if there is a single equation for the whole variety which is of polynomially bounded degree. We do manage to reverse the order of quantifiers and prove such an upper bound in Theorem 1. This suggests that the set of non-rigid matrices is much less complex than what one may suspect given the results of [11, 8].

## 1.3 Circuits for linear transformations

The original motivation for defining rigidity was in the context of proving lower bounds for algebraic circuits [18]. If  $A \in \mathbb{F}^{n \times n}$  is an  $(\varepsilon n, n^{1+\delta})$ -rigid matrix, for any  $\varepsilon, \delta > 0$ , then the linear transformation represented by  $A$  cannot be computed by an algebraic circuit of depth  $O(\log n)$  and size  $O(n)$ .

Every algebraic circuit computing a linear transformation is without loss of generality a *linear* circuit. A linear circuit is a directed acyclic graph that has  $n$  inputs labeled  $X_1, \dots, X_n$  and  $n$  output nodes. Each edge is labeled by a scalar  $\alpha \in \mathbb{F}$ . Each node computes a linear

function in  $X_1, \dots, X_n$  defined inductively. An internal node  $u$  with children,  $v_1, \dots, v_k$ , connected to it by edges labeled  $\alpha_1, \dots, \alpha_k$ , computes the linear function  $\sum_i \alpha_i \ell_{v_i}$ , where  $\ell_{v_i}$  is the linear function computed by  $v_i$ ,  $1 \leq i \leq k$ . The size of the circuit is the number of edges in the circuit.

It is possible to use similar techniques to those used in the proof of Theorem 1 in order to prove a polynomial upper bound on an equation for a variety containing all matrices  $A \in \mathbb{F}^{n \times n}$  whose corresponding linear transformation can be computed by an algebraic circuit of size at most  $n^2/200$  (even without restriction on the depth). Note that this is nearly optimal as any such linear transformation can be computed by a circuit of size  $n^2$ . More formally, we show the following.

► **Theorem 2.** *Let  $\mathbb{F}$  be a field of size at least  $n^2$ . For every large enough  $n$ , there exists a non-zero polynomial  $Q \in \mathbb{F}[x_{1,1}, \dots, x_{n,n}]$ , of degree at most  $n^3$ , which is a non-trivial equation for matrices which are computed by algebraic circuit of size at most  $n^2/200$ .*

Our proofs are based on a dimension counting arguments, and are therefore non-constructive and do not give explicit equations for the relevant varieties. It thus remains a very interesting open problem to provide explicit low-degree equations for any of the varieties considered in this paper. Here “explicit” means a polynomial which has arithmetic circuits of size  $\text{poly}(n)$ .<sup>3</sup> The question of whether such equations exists has a win-win flavor: if they do, this can aid in explicit constructions of rigid matrices, and on the other hand, if all equations are hard, we have identified a family of polynomials which requires super-polynomial arithmetic circuits. Assuming the existence of a polynomial time algorithm for polynomial identity testing, we are able to make this connection formal.

Let PIT denote the set of strings which describe arithmetic circuits (say, over  $\mathbb{C}$ ) which compute the zero polynomial. It is well known that  $\text{PIT} \in \text{coRP}$ . Kabanets and Impagliazzo [10] proved that certain circuit lower bounds follow from the assumption that  $\text{PIT} \in \text{P}$ . As a corollary to Theorem 2, we are able to prove theorem of a similar kind.

► **Corollary 3.** *Suppose  $\text{PIT} \in \text{P}$ . Then at least one of the following is true:*

1. *There exists a family of  $n$ -variate polynomials of degree  $\text{poly}(n)$  over  $\mathbb{C}$ , which can be computed (as its list of coefficients, given the input  $1^n$ ) in PSPACE, which does not have polynomial size constant free arithmetic circuits.*
2. *there exists a family of matrices, constructible in polynomial time with an NP oracle (given the input  $1^n$ ), which requires linear circuits of size  $\Omega(n^2)$ .*

A *constant free arithmetic circuit* is an arithmetic circuit which is only allowed to use the constants  $\{0, \pm 1\}$ .

A different way to interpret Corollary 3 is as saying that at least one of the following three lower bound results hold: either  $\text{PIT} \notin \text{P}$ , or (at least) one of the two circuit lower bounds stated in the corollary. We emphasize that the result holds under *any* (even so-called *white box*) derandomization of PIT.

Our statement is similar to, but incomparable with the result of Kabanets and Impagliazzo [10] who proved that if  $\text{PIT} \in \text{P}$  then either the permanent does not have polynomial size constant free arithmetic circuits, or  $\text{NEXP} \not\subseteq \text{P}/\text{poly}$ .

Since  $(\varepsilon n, \varepsilon n^2)$ -rigid matrices have linear circuit of size  $3\varepsilon n^2$ , the last item of Corollary 3 in particular implies a conditional construction of  $(\Omega(n), \Omega(n^2))$ -rigid matrices (it is also possible to directly use Theorem 1 instead of Theorem 2 to deduce this result). Unconditional

<sup>3</sup> Although one may consider other, informal notions of explicitness which could nevertheless be helpful.

constructions of rigid matrices in polynomial time with an NP oracle were recently given in [2, 3]. However, the rigidity parameters in these papers are not enough to imply circuit lower bounds (furthermore, even optimal rigidity parameters are not enough to imply  $\Omega(n^2)$  lower bounds for general linear circuits).

Since it is widely believed that  $\text{PIT} \in \text{P}$ , the answer to which of the last two items of Corollary 3 holds boils down to the question of whether there exists an equation for non-rigid matrices of degree  $\text{poly}(n)$  and circuit size  $\text{poly}(n)$ . If determining if a matrix is rigid is  $\text{coNP}$ -hard (as is known for some restricted ranges of parameters [13]), it is tempting to also believe that the equations should not be easily computable, as they provide “proof” for rigidity which can be verified in randomized polynomial time. However, it could still be the case that those equations that have polynomial size circuits only prove the rigidity of “easy” instances.

## 1.4 Some basic notions in algebraic geometry

For completeness, in this section we define some basic notions in algebraic geometry. A reader who is familiar with this topic may skip to the next section.

Let  $\mathbb{F}$  be an algebraically closed field. A set  $V \subseteq \mathbb{F}^n$  is called an *affine variety* if there exist polynomials  $f_1, \dots, f_t \in \mathbb{F}[x_1, \dots, x_n]$  such that  $V = \{\mathbf{x} : f_1(\mathbf{x}) = f_2(\mathbf{x}) = \dots = f_t(\mathbf{x}) = 0\}$ . For convenience, in this paper we often refer to affine varieties simply as varieties.

For each variety  $V$  there is a corresponding ideal  $\mathbf{I}(V) \subseteq \mathbb{F}[x_1, \dots, x_n]$  which is defined as

$$\mathbf{I}(V) := \{f \in \mathbb{F}[x_1, \dots, x_n] : f(\mathbf{x}) = 0 \text{ for all } \mathbf{x} \in V\}.$$

Conversely, for an ideal  $I \subseteq \mathbb{F}[x_1, \dots, x_n]$  we may define the variety

$$\mathbf{V}(I) = \{\mathbf{x} : f(\mathbf{x}) = 0 \text{ for all } f \in I\}.$$

Given a set  $A \subseteq \mathbb{F}^n$  we may similarly define the ideal  $\mathbf{I}(A)$ . The (Zariski) *closure* of a set  $A$ , denoted  $\overline{A}$ , is the set  $\mathbf{V}(\mathbf{I}(A))$ . In words, the closure of  $A$  is the set of common zeros of all the polynomials that vanish on  $A$ . It is also the smallest variety with respect to inclusion which contains  $A$ . By construction,  $\overline{A}$  is a variety, and a polynomial which vanishes everywhere on  $A$  is also vanishes on  $\overline{A}$ .

Over  $\mathbb{C}$ , it is instructive to think of the Zariski closure of  $A$  as the usual Euclidean closure. In fact, for the various sets  $A$  we consider in this paper (which correspond to sets of “low complexity” objects, e.g., non-rigid matrices or matrices which can be computed with a small circuit), it can be shown that these two notions of closure coincide (see, e.g., Section 4.2 of [4]).

A variety  $V$  is called *irreducible* if it cannot be written as a union  $V = V_1 \cup V_2$  of varieties  $V_1, V_2$  that are properly contained in  $V$ . Every variety can be uniquely written as a union  $V = V_1 \cup V_2 \cup \dots \cup V_m$  of irreducible varieties. The varieties  $V_1, \dots, V_m$  are then called the *irreducible components* of  $V$ .

## 2 Degree Upper Bound for Non-Rigid Matrices

In this section, we prove Theorem 1. A key component of the proof is the use of the following construction, due to Shpilka and Volkovich, which provides an explicit low-degree polynomial map on a small number of variables, which contains all sparse matrices in its image. For completeness, we provide the construction and prove its basic property.

► **Lemma 4** ([17]). *Let  $\mathbb{F}$  be a field such that  $|\mathbb{F}| > n$ . Then for all  $k \in \mathbb{N}$ , there exists an explicit polynomial map  $\text{SV}_{n,k}(\mathbf{x}, \mathbf{y}) : \mathbb{F}^{2k} \rightarrow \mathbb{F}^n$  of degree at most  $n$  such that for any subset  $T = \{i_1, \dots, i_k\} \subseteq [n]$  of size  $k$ , there exists a setting  $\mathbf{y} = \boldsymbol{\alpha}$  such that  $\text{SV}(\mathbf{x}, \boldsymbol{\alpha})$  is identically zero on every coordinate  $j \notin T$ , and equals  $x_j$  in coordinate  $i_j$  for all  $j \in [k]$ .*

**Proof.** Arbitrarily pick distinct  $\alpha_1, \dots, \alpha_n \in \mathbb{F}$ , and let  $u_1, \dots, u_n$  be their corresponding Lagrange's interpolation polynomials, i.e., polynomials of degree at most  $n-1$  such that  $u_i(\alpha_j) = 1$  if  $j = i$  and 0 otherwise (more explicitly,  $u_i(z) = \frac{\prod_{j \neq i} (z - \alpha_j)}{\prod_{j \neq i} (\alpha_i - \alpha_j)}$ ).

Let  $P_i(x_1, \dots, x_k, y_1, \dots, y_k) = \sum_{j=1}^k u_i(y_j) \cdot x_j$ , and finally let

$$\text{SV}_{n,k}(\mathbf{x}, \mathbf{y}) = (P_1(\mathbf{x}, \mathbf{y}), \dots, P_n(\mathbf{x}, \mathbf{y})).$$

It readily follows that given  $T = \{i_1, \dots, i_k\}$  as in the statement of the lemma, we can set  $y_j = \alpha_{i_j}$  for  $j \in [k]$  to derive the desired conclusion. The upper bound on the degree follows by inspection. ◀

As a step toward the proof of Theorem 1, we show there is a polynomial map on much fewer than  $n^2$  variables with degree polynomially bounded in  $n$  such that its image contains every non-rigid matrix. In the next step, we show that the image of every such polynomial map has an equation of degree  $\text{poly}(n)$ .

► **Lemma 5.** *There exists an explicit polynomial map  $P : \mathbb{F}^{4\epsilon n^2} \rightarrow \mathbb{F}^{n \times n}$ , of degree at most  $n^2$ , such that every matrix  $M$  which is not  $(\epsilon n, \epsilon n^2)$  rigid lies in its image.*

**Proof.** Let  $k = \epsilon n^2$  and let  $\mathbf{u}, \mathbf{v}, \mathbf{x}, \mathbf{y}$  denote disjoint tuples of  $k$  variables each.

Let  $U$  be a symbolic  $n \times \epsilon n$  matrix whose entries are labeled by the variables  $\mathbf{u}$ , and similarly let  $V$  be a symbolic  $\epsilon n \times n$  matrix labeled by  $\mathbf{v}$ . Let  $UV(\mathbf{u}, \mathbf{v}) : \mathbb{F}^{2k} \rightarrow \mathbb{F}^{n \times n}$  be the degree 2 polynomial map defined by the matrix multiplication  $UV$ .

Finally, let  $P : \mathbb{F}^{4k} \rightarrow \mathbb{F}^{n \times n}$  be defined as

$$P(\mathbf{u}, \mathbf{v}, \mathbf{x}, \mathbf{y}) = UV(\mathbf{u}, \mathbf{v}) + \text{SV}_{n^2,k}(\mathbf{x}, \mathbf{y}),$$

where  $\text{SV}_{n^2,k}$  is as defined in Lemma 4.

Suppose now  $M$  is a non-rigid matrix, i.e.,  $M = R + S$  for  $R$  of rank  $\epsilon n$  and  $S$  which is  $\epsilon n^2$ -sparse. Decompose  $R = U_0 V_0$  for  $n \times \epsilon n$  matrix  $U_0$  and  $\epsilon n \times n$  matrix  $V_0$ . Let  $T$  denote the support of  $S$ . For convenience we may assume  $|T| = k$  (otherwise, pad with zeros arbitrarily). Let  $\boldsymbol{\alpha} \in \mathbb{F}^k$  denote the setting for  $\mathbf{y}$  in  $\text{SV}_{n^2,k}$  which maps  $x_1, \dots, x_k$  to  $T$ , and let  $\mathbf{s} = (s_1, \dots, s_k)$  denote the non-zero entries of  $S$ . Then

$$P(U_0, V_0, \mathbf{s}, \boldsymbol{\alpha}) = U_0 V_0 + S = R + S = M. \quad \blacktriangleleft$$

To complete the proof of Theorem 1, we now argue that the image of any polynomial map with parameters as in Lemma 5 has an equation of degree at most  $n^3$ .

**Proof of Theorem 1.** Let  $V_1$  denote the subspace of polynomials over  $\mathbb{F}$  in  $n^2$  variables of degree at most  $n^3$ . Let  $V_2$  denote the subspace of polynomials over  $\mathbb{F}$  in  $4\epsilon n^2$  variables of degree at most  $n^5$ . Let  $P$  be as in Lemma 5, and consider the linear transformation  $T : V_1 \rightarrow V_2$  given by  $Q \mapsto Q \circ P$ , where  $Q \circ P$  denotes the composition of the polynomial  $Q$  with the map  $P$ , i.e.,  $(Q \circ P)(\mathbf{x}) = Q(P(\mathbf{x}))$  (indeed, observe that since  $\deg(Q) \leq n^3$  and  $\deg(P) \leq n^2$ , it follows that  $\deg(Q \circ P) \leq n^5$ ).



We have that  $\dim(V_1) = \binom{n^3+n^2}{n^2} \geq n^{n^2}$ , whereas  $\dim(V_2) = \binom{4\epsilon n^2+n^5}{4\epsilon n^2} \leq (2n^5)^{4\epsilon n^2} < \dim(V_1)$  by the choice of  $\epsilon$ , so that there exists a non-zero polynomial in the kernel of  $T$ , that is,  $0 \neq Q_0 \in V_1$  such that  $Q_0 \circ P \equiv 0$ .

It remains to be shown that for any non-rigid matrix  $M$ ,  $Q_0(M) = 0$ . Indeed, let  $M$  be a non-rigid matrix. By Lemma 5, there exist  $\beta \in \mathbb{F}^{4\epsilon n^2}$  such that  $P(\beta) = M$ . Thus,  $Q_0(M) = Q_0(P(\beta)) = Q_0 \circ P(\beta) = 0$ , as  $Q_0 \circ P \equiv 0$ .  $\blacktriangleleft$

► **Remark 6.** If the support of the sparse matrix is fixed a-priori to some set  $S \subseteq [n] \times [n]$  of cardinality at most  $\epsilon n^2$ , then it is easier to come up with a universal map  $\tilde{P}$  from  $\mathbb{F}^{3\epsilon n^2} \mapsto \mathbb{F}^{n \times n}$  such that every matrix  $M$  whose rank can be reduced to at most  $\epsilon n$  by changing entries in the set  $S$  is contained in the image of  $\tilde{P}$ . Just consider  $\tilde{P}(\mathbf{w}, \mathbf{x}, \mathbf{y}) = UV(\mathbf{u}, \mathbf{v}) + W$ , where  $W$  is a matrix such that for all  $(i, j) \in [n] \times [n]$ , if  $(i, j) \in S$ , then  $W(i, j) = w_{i,j}$  and  $W(i, j)$  is zero otherwise. Here, each  $w_{i,j}$  is a distinct formal variable. Combined with the dimension comparison argument we used in the proof of Theorem 1, it can be seen that there is a non-zero low degree polynomial  $\tilde{Q}$  such that  $\tilde{Q} \circ \tilde{P} \equiv 0$ . This argument provides a (different) equation of polynomial degree for each irreducible component of the variety of non-rigid matrices.

► **Remark 7.** It is possible to use the equation given in Theorem 1, and using the methods of [11], to construct “semi-explicit”  $(\epsilon n, \epsilon n^2)$ -rigid matrices. These are matrices whose entries are algebraic numbers (over  $\mathbb{Q}$ ) with short description, which are non-explicit from the computational complexity point of view. However, such constructions are also known using different methods (see Section 2.4 of [12]).

### 3 Degree Upper bound for Other Models

The proof of Theorem 2 is omitted from this version and appears in the full version of the paper. The strategy, as before, is to observe that all matrices with a small circuit lie in the image of a polynomial map  $P$  on a small number of variables and small degree. Circuits of size  $s$  can have many different topologies and thus we first construct a “universal” linear circuit, of size  $s' \leq s^4$ , that contains as subcircuits all linear circuits of size  $s$ . Importantly,  $s'$  will affect the degree of  $P$  but not its number of variables. We note that this construction of universal circuits is slightly different from similar constructions in earlier work, e.g., in [16]; the key difference being that a naive use of ideas in [16] to obtain the map  $P$  seems to incur an asymptotic increase in the number of variables of  $P$ , which is unacceptable in our current setting.

Analogous to the proof of Theorem 1, we then observe via a dimension counting argument that the image of the polynomial map  $P$  has an equation of degree at most  $n^3$ . This completes the proof of Theorem 2.

Another algebraic object which is closely related to proving circuit lower bounds is the set of three dimensional tensors of high rank. A three dimensional tensor of rank at least  $r$  implies a lower bound of  $r$  on an arithmetic circuit computing the bi-linear function associated with the tensor. Our arguments also provide polynomial degree upper bounds for the set of tensors of (border) rank at most  $n^2/300$ . We again omit the details, which appear in the full version of the paper.

A similar method can be used to prove the existence of an equation of degree  $\text{poly}(n)$  for three dimensional tensors of *slice rank* (see, e.g., [5]) at most, say,  $n/1000$ . The existence of such an equation was proved (using different techniques) in [5].



## 4 Applications to Circuit Lower Bounds

In this section we prove Corollary 3. The strategy of the proof is simple: the proof of Theorem 2 implies a PSPACE algorithm which produces a sequence of polynomials which are equations for the set of matrices with small linear circuits. If those equations require large circuits, we are done, and if not, then there exists an equation with small circuits which (assuming  $\text{PIT} \in \text{P}$ ) can be found using an NP-oracle. Using, once again, the assumption that  $\text{PIT} \in \text{P}$ , we can also find deterministically a matrix on which the equation evaluates to non-zero, which implies the matrix requires large linear circuits.

There are some technical difficulties involved with this plan which we now describe. The first problem is that even arithmetic circuits of small size can have large description as bit strings, due to the field constants appearing in the circuits. To prevent this issue, we only consider *constant free* arithmetic circuits, which are only allowed inputs labeled by  $\{0, \pm 1\}$  (but can still compute other constants in the circuit using arithmetic operations).

The second problem is that, in order to be able to find a non-zero of the equation in the last step of the algorithm (using the mere assumption that  $\text{PIT} \in \text{P}$ ), we need not only the size of the circuit but also its *degree* to be bounded by  $\text{poly}(n)$ . Of course, by Theorem 2 there exists such a circuit, but we need to be able to prevent a malicious prover from providing us with a  $\text{poly}(n)$  size circuit of exponential degree, and it is not known how to compute the degree of a circuit in deterministic polynomial time, even assuming  $\text{PIT} \in \text{P}$ . To solve this issue, we use an idea of Malod and Portier [14], who showed that any polynomial with circuit of size  $\text{poly}(n)$  and degree  $d$  also has a *multiplicatively disjoint* (MD) circuit of size  $\text{poly}(n, d)$ . An MD circuit is a circuit in which any multiplication gate multiplies two disjoint subcircuits. This is a syntactic notion which is easy to verify efficiently and deterministically, and an MD circuit of size  $s$  is guaranteed to compute a polynomial of degree at most  $s$ .

A final technical issue is that the notion of MD circuits does not fit perfectly within the framework of constant free circuits. Therefore we use the notion of “almost MD” circuits, which allow for the case which the inputs to a multiplication gates are not disjoint, as long as at least one of them is the root of a subcircuit in which only constants appear.

► **Definition 8.** We say a gate  $v$  in a circuit is *constant producing (CP)* if in the subcircuit rooted at  $v$ , all input nodes are field constants.

An *almost-MD circuit* is a circuit where every multiplication gate either multiplies two disjoint subcircuits, or at least one of its children is constant producing.

► **Lemma 9.** Suppose  $f$  is an  $n$ -variate polynomial of degree  $\text{poly}(n)$  which has a constant free arithmetic circuit of degree  $\text{poly}(n)$ . Then  $f$  has a constant free almost-MD circuit of size  $\text{poly}(n)$ .

The proof of Lemma 9 appears in the full version of the paper.

For circuits which compute low-degree polynomials, the mere existence of an algorithm for the decision version of PIT allows one to construct an algorithm for the search version. The following lemma is standard, and its proof appears in the full version of the paper.

► **Lemma 10.** Suppose  $\text{PIT} \in \text{P}$ . Then there is a polynomial time algorithm that given a non-zero almost-MD arithmetic circuit  $C$  of size  $s$  computing an  $n$ -variate polynomial, finds in time  $\text{poly}(n, s)$  an element  $\mathbf{a} \in \mathbb{C}^n$  such that  $C(\mathbf{a}) \neq 0$ .

As we noted above, the assumption that  $C$  is almost-MD was used in Lemma 10 to bound the degree of the circuit. It is also useful because it is easy to decide in deterministic polynomial time whether a circuit is almost-MD. We now complete the proof of Corollary 3.

**Proof of Corollary 3.** For every  $n$ , the proof of Theorem 2 provides an equation  $Q_n$  for the set of  $n \times n$  matrices with small linear circuits. This polynomial can be found by solving a linear system of equations in a linear space whose dimension is  $\exp(\text{poly}(n))$ . Using standard, small space algorithm for linear algebra [6, 1], this implies that there exists a fixed PSPACE algorithm which, on input  $1^n$ , outputs the list of coefficients of the polynomial  $Q_n$ .

Consider now the family  $\{Q_n\}_{n \in \mathbb{N}}$ . If for any constant  $k \in \mathbb{N}$  there exist infinitely many  $n \in \mathbb{N}$  such that  $Q_n$  requires circuits of size at least  $n^k$ , it follows (by definition) that the PSPACE algorithm above outputs a family of polynomials with super-polynomial constant-free arithmetic circuits.

We are thus left to consider the case that there exists a constant  $k \in \mathbb{N}$  such that for all large enough  $n \in \mathbb{N}$ ,  $Q_n$  can be computed by circuits of size  $n^k$ . By Lemma 9, we may assume without loss of generality that these circuits are almost-MD circuits. Further suppose  $\text{PIT} \in \text{P}$ . We will show how to construct a matrix in polynomial time with an NP oracle which requires large linear circuits.

Consider the language  $L$  of pairs  $(1^n, x)$  such that there exists a string  $y$  of length at most  $n^k$  such that  $xy$  describes an almost-MD circuit  $C$  such that  $C$  is non-zero, and  $C \circ U \equiv 0$ , where  $U$  is the polynomial map given in the proof of Theorem 2.

Assuming  $\text{PIT} \in \text{P}$ , the language  $L$  is in NP, and by assumption for every large enough  $n$  there exists such a circuit. Thus, we can use the NP oracle to construct such a circuit  $C$  bit by bit. Finally, using Lemma 10 we can output a matrix  $M$  such that  $C(M) \neq 0$ .

By the properties of the circuit  $C$  and the map  $U$ ,  $M$  does not have linear circuits of size less than  $n^2/200$ . ◀

Many variations of Corollary 3 can be proved as well, with virtually the same proof. By slightly modifying the language  $L$  used in the proof, it is possible to prove the same result even under the assumption  $\text{PIT} \in \text{NP}$  (recall that  $\text{PIT} \in \text{coRP}$ ). A similar statements also holds over finite fields of size  $\text{poly}(n)$ , in which case the proof is simpler since there are no issues related to the bit complexity of the first constants. Finally, an analog of Corollary 3 also holds for tensor rank: that is, assuming  $\text{PIT} \in \text{P}$ , either there exists a construction of a hard polynomial in PSPACE, or an efficient construction with an NP oracle of a 3-dimensional tensor of rank  $\Omega(n^2)$ . We remark that for tensors of large rank there are no analogs of [2, 3], i.e., there do not exist even constructions with an NP oracle of tensors with slightly super-linear rank.

---

## References

- 1 Eric Allender, Robert Beals, and Mitsunori Ogihara. The complexity of matrix rank and feasible systems of linear equations. *Comput. Complex.*, 8(2):99–126, 1999. doi:10.1007/s000370050023.
- 2 Josh Alman and Lijie Chen. Efficient construction of rigid matrices using an NP oracle. In *Proceedings of the 60th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2019)*, pages 1034–1055. IEEE Computer Society, 2019. doi:10.1109/FOCS.2019.00067.
- 3 Amey Bhangale, Prahladh Harsha, Orr Paradise, and Avishay Tal. Rigid matrices from rectangular pcps. *Electronic Colloquium on Computational Complexity (ECCC)*, 27:75, 2020. URL: <https://eccc.weizmann.ac.il/report/2020/075>.
- 4 Markus Bläser and Christian Ikenmeyer. Introduction to geometric complexity theory. Lecture notes, 2017. URL: [http://pcwww.liv.ac.uk/~iken/teaching\\_sb/summer17/introtoget/gct.pdf](http://pcwww.liv.ac.uk/~iken/teaching_sb/summer17/introtoget/gct.pdf).

- 5 Markus Bläser, Christian Ikenmeyer, Vladimir Lysikov, Anurag Pandey, and Frank-Olaf Schreyer. Variety membership testing, algebraic natural proofs, and geometric complexity theory. *CoRR*, abs/1911.02534, 2019. [arXiv:1911.02534](#).
- 6 Allan Borodin, Joachim von zur Gathen, and John E. Hopcroft. Fast parallel matrix and GCD computations. *Inf. Control.*, 52(3):241–256, 1982. doi:10.1016/S0019-9958(82)90766-5.
- 7 Michael A. Forbes, Amir Shpilka, and Ben Lee Volk. Succinct hitting sets and barriers to proving lower bounds for algebraic circuits. *Theory of Computing*, 14(1):1–45, 2018. doi:10.4086/toc.2018.v014a018.
- 8 Fulvio Gesmundo, Jonathan D. Hauenstein, Christian Ikenmeyer, and J. M. Landsberg. Complexity of linear circuits and geometry. *Foundations of Computational Mathematics*, 16(3):599–635, 2016. doi:10.1007/s10208-015-9258-8.
- 9 Joshua A. Grochow, Mrinal Kumar, Michael E. Saks, and Shubhangi Saraf. Towards an algebraic natural proofs barrier via polynomial identity testing. *CoRR*, abs/1701.01717, 2017. [arXiv:1701.01717](#).
- 10 Valentine Kabanets and Russell Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Computational Complexity*, 13(1-2):1–46, 2004. Preliminary version in the *35th Annual ACM Symposium on Theory of Computing (STOC 2003)*. doi:10.1007/s00037-004-0182-6.
- 11 Abhinav Kumar, Satyanarayana V. Lokam, Vijay M. Patankar, and Jayalal Sarma. Using elimination theory to construct rigid matrices. *Computational Complexity*, 23(4):531–563, 2014. doi:10.1007/s00037-013-0061-0.
- 12 Satyanarayana V. Lokam. Complexity lower bounds using linear algebra. *Foundations and Trends in Theoretical Computer Science*, 4(1-2):1–155, 2009. doi:10.1561/04000000011.
- 13 Meena Mahajan and Jayalal Sarma. On the complexity of matrix rank and rigidity. *Theory Comput. Syst.*, 46(1):9–26, 2010. doi:10.1007/s00224-008-9136-8.
- 14 Guillaume Malod and Natacha Portier. Characterizing valiant’s algebraic complexity classes. *J. Complex.*, 24(1):16–38, 2008. doi:10.1016/j.jco.2006.09.006.
- 15 Ketan Mulmuley and Milind A. Sohoni. Geometric complexity theory I: an approach to the P vs. NP and related problems. *SIAM J. Comput.*, 31(2):496–526, 2001. doi:10.1137/S009753970038715X.
- 16 Ran Raz. Elusive functions and lower bounds for arithmetic circuits. *Theory of Computing*, 6(7):135–177, 2010. doi:10.4086/toc.2010.v006a007.
- 17 Amir Shpilka and Ilya Volkovich. Read-once polynomial identity testing. *Computational Complexity*, 24(3):477–532, 2015. Preliminary version in the *40th Annual ACM Symposium on Theory of Computing (STOC 2008)*. doi:10.1007/s00037-015-0105-8.
- 18 Leslie G. Valiant. Graph-theoretic arguments in low-level complexity. In Jozef Gruska, editor, *Proceedings of the 2nd International Symposium on the Mathematical Foundations of Computer Science (MFCS 1977)*, volume 53 of *Lecture Notes in Computer Science*, pages 162–176. Springer, 1977. doi:10.1007/3-540-08353-7\_135.



# The Strongish Planted Clique Hypothesis and Its Consequences

**Pasin Manurangsi**

Google Research, Mountain View, CA, USA  
pasin@google.com

**Aviad Rubinfeld**

Stanford University, CA, USA  
aviad@cs.stanford.edu

**Tselil Schramm**

Stanford University, CA, USA  
tselil@stanford.edu

---

## Abstract

We formulate a new hardness assumption, the *Strongish Planted Clique Hypothesis (SPCH)*, which postulates that any algorithm for planted clique must run in time  $n^{\Omega(\log n)}$  (so that the state-of-the-art running time of  $n^{O(\log n)}$  is optimal up to a constant in the exponent).

We provide two sets of applications of the new hypothesis. First, we show that SPCH implies (nearly) tight inapproximability results for the following well-studied problems in terms of the parameter  $k$ : Densest  $k$ -Subgraph, Smallest  $k$ -Edge Subgraph, Densest  $k$ -Subhypergraph, Steiner  $k$ -Forest, and Directed Steiner Network with  $k$  terminal pairs. For example, we show, under SPCH, that no polynomial time algorithm achieves  $o(k)$ -approximation for Densest  $k$ -Subgraph. This inapproximability ratio improves upon the previous best  $k^{o(1)}$  factor from (Chalermsook et al., FOCS 2017). Furthermore, our lower bounds hold even against fixed-parameter tractable algorithms with parameter  $k$ .

Our second application focuses on the complexity of graph pattern detection. For both induced and non-induced graph pattern detection, we prove hardness results under SPCH, improving the running time lower bounds obtained by (Dalirrooyfard et al., STOC 2019) under the Exponential Time Hypothesis.

**2012 ACM Subject Classification** Theory of computation → Problems, reductions and completeness; Theory of computation → Fixed parameter tractability

**Keywords and phrases** Planted Clique, Densest  $k$ -Subgraph, Hardness of Approximation

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.10

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/2011.05555>.

**Funding** *Aviad Rubinfeld*: A.R. is supported in part by a Packard Fellowship.

*Tselil Schramm*: T.S. is supported by a Simons Institute Microsoft Research Fellowship.

**Acknowledgements** Pasin would like to thank Michal Pilipczuk and Daniel Lokshtanov for posing the approximability of Densest  $k$ -Subhypergraph as an open problem at Dagstuhl Seminar on New Horizons in Parameterized Complexity, and for helpful discussions.

## 1 Introduction

The last couple of decades have seen dramatic advances in our understanding of parameterized, fine-grained, and average-case complexity. To a large extent, this progress has been enabled by bolder computational hardness assumptions, beyond the classical  $P \neq NP$ . Two notable assumptions in these fields are the Exponential Time Hypothesis and the Planted Clique Hypothesis. In this paper we propose a new hypothesis, the *Strongish Planted Clique*



© Pasin Manurangsi, Aviad Rubinfeld, and Tselil Schramm;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 10; pp. 10:1–10:21



Leibniz International Proceedings in Informatics

LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

*Hypothesis*, which strengthens the Planted Clique Hypothesis in the style of the Exponential Time Hypothesis. We show that this hypothesis has interesting implications in the parameterized complexity of both approximation problems and graph pattern detection.

**Exponential Time Hypothesis.** The Exponential Time Hypothesis (ETH) [69] is a pessimistic version of  $P \neq NP$  which postulates that solving 3-SAT on  $n$  variables requires time  $2^{\Omega(n)}$ . In other words, the running times of the state-of-the-art (and the brute-force) algorithms for 3-SAT are optimal up to a constant factor in the exponent. ETH has important applications in parameterized complexity (e.g. [30, 82, 80, 46, 48]) and hardness of approximation (e.g. [26, 25, 92, 84, 87, 52]). In the past several years, further progress was achieved by assuming stronger variants of ETH such as the Strong ETH (SETH) in fine-grained complexity [70, 95], Gap-ETH in parameterized complexity [31, 23, 83], and ETH for PPAD in Algorithmic Game Theory [9, 91].

**Planted Clique Hypothesis.** In the Planted  $\kappa$ -Clique Problem, the goal is to distinguish (with high probability) between graphs sampled from one of the following distributions: Uniformly at random<sup>1</sup>; and uniformly at random, with an added  $\kappa$ -clique. While statistically it is easy to distinguish the two distributions for  $\kappa$  as little as  $2.1 \log(n)$ , the *Planted Clique Hypothesis* (PCH) postulates that no polynomial time algorithm can solve this problem, even for  $\kappa$  as large as  $o(\sqrt{n})$ . The history of this problem goes back to Karp [73] and Jerrum [71], and in the past decade it has been popular as a hardness assumption for both worst-case [8, 68, 3, 10, 20] and average-case [17, 66, 18, 63, 29, 27] problems. A simple  $n^{\Theta(\log(n))}$ -time algorithm for the planted- $\kappa$ -clique problem non-deterministically guesses  $\ell = \Theta(\log(n))$  vertices from the clique, and then checks whether all of their common neighbors form a clique. There are several other algorithms that also solve this problem in time  $n^{\Theta(\log(n))}$  [60, 59, 86, 13], but no faster algorithm is known for  $\kappa = O(n^{0.49})$ .

## Strongish Planted Clique Hypothesis

In analogy to the Exponential Time Hypothesis for 3-SAT, we propose the following hypothesis, which postulates that the state-of-the-art algorithms for the Planted Clique Problem are optimal up to a constant factor in the exponent. A *Strong* Planted Clique Hypothesis, in analogy with SETH, would specify a precise constant in the exponent – our hypothesis is merely Strong-ish. We let  $\mathcal{G}(n, p)$  denote the Erdős-Rényi distribution with parameter  $p$ , and  $\mathcal{G}(n, p, \kappa)$  denote the Erdős-Rényi distribution with a planted  $\kappa$ -clique.

► **Hypothesis 1** (Strongish Planted Clique Hypothesis (SPCH)). *There exists a constant  $\delta \in (0, \frac{1}{2})$  such that no  $n^{o(\log n)}$ -time algorithm  $\mathcal{A}$  satisfies both of the following:*

- (Completeness)  $\Pr_{G \sim \mathcal{G}(n, \frac{1}{2}, \lceil n^\delta \rceil)}[\mathcal{A}(G) = 1] \geq 2/3$ .
- (Soundness)  $\Pr_{G \sim \mathcal{G}(n, \frac{1}{2})}[\mathcal{A}(G) = 1] \leq 1/3$ .

In addition to the lack of algorithmic progress toward refuting this hypothesis, we note that  $n^{\Theta(\log(n))}$  is in fact provably optimal for the Sum-of-Squares hierarchy [11], which captures the state-of-the-art algorithmic techniques for a number of average-case problems. It is also known to be tight for statistical algorithms [62, 28].

<sup>1</sup> I.e. from the *Erdős-Rényi* (ER) distribution over  $n$ -vertex graphs where each edge appears independently with probability  $1/2$ .

## 1.1 Our Contributions: Hardness results from Strongish Planted Clique Hypothesis

Our main technical contributions are in exploring the implications of our new SPCH in parameterized complexity. We prove two types of hardness results: hardness of approximation, and hardness of (exact) graph pattern detection. Due to the rich literature for each problem we consider, we will only mention the most relevant results here and defer a more comprehensive discussion to Section 1.4.

### 1.1.1 Hardness of approximation from SPCH

At the heart of our work is the study of the *Densest  $k$ -Subgraph* problem, in which we are given an undirected graph  $G = (V, E)$  and a positive integer  $k$ . The goal is to output a subset  $S \subseteq V$  of  $k$  vertices that induces as many edges as possible. There is a trivial  $O(k)$ -approximation for the problem: return an arbitrary set of  $\lfloor k/2 \rfloor$  edges. We show that assuming SPCH, this algorithm is optimal:

► **Theorem 2.** *Assuming the Strongish Planted Clique Hypothesis (Hypothesis 1), there is no  $f(k) \cdot \text{poly}(n)$ -time algorithm that can approximate Densest  $k$ -Subgraph on  $n$ -vertex graphs to within a factor  $o(k)$  for any function  $f$ . Furthermore, this holds even in the perfect completeness case where the input graph is promised to contain a  $k$ -clique.*

Theorem 2 improves upon the inapproximability ratio of  $k^{o(1)}$  shown in [31] under Gap-ETH.

The approximability of Densest  $k$ -Subgraph is known to be intimately related to that of numerous other problems. As such, our tight hardness of approximation for Densest  $k$ -Subgraph immediately implies several tight approximability results as corollaries, which we list below.

- *Smallest  $k$ -Edge Subgraph*: given an undirected graph  $G = (V, E)$  and a positive integer  $k$ , find a smallest subset  $S \subseteq V$  that induces at least  $k$  edges. For this problem, the trivial solution that chooses  $k$  edges arbitrarily is an  $O(\sqrt{k})$ -approximation since even the optimum requires at least  $\sqrt{k}$  vertices. We show that this is tight (Corollary 11): no fixed-parameter tractable (FPT) (in  $k$ ) algorithm can achieve  $o(\sqrt{k})$  approximation ratio.
- *Steiner  $k$ -Forest* (aka  *$k$ -Forest*): given an edge-weighted undirected graph  $G = (V, E)$ , a set  $\{(s_1, t_1), \dots, (s_\ell, t_\ell)\}$  of demand pairs and a positive integer  $k$ , the goal is to find a (not necessarily induced) subgraph of  $G$  with smallest total edge weight that connects at least  $k$  demand pairs. In Corollary 12, we show that no FPT (in  $k$ ) algorithm can achieve  $o(\sqrt{k})$  approximation ratio. This matches the  $O(\sqrt{k})$ -approximation algorithm by Gupta et al. [65].
- *Directed Steiner Network* (aka *Directed Steiner Forest*): given an edge-weighted directed graph  $G = (V, E)$  and a set  $\{(s_1, t_1), \dots, (s_k, t_k)\}$  of  $k$  demand pairs, the goal is to find a (not necessarily induced) subgraph of  $G$  with smallest total edge weight in which  $t_i$  is reachable from  $s_i$  for all  $i = 1, \dots, k$ . We prove that no FPT (in  $k$ ) algorithm achieves  $o(\sqrt{k})$ -approximation (Corollary 14). This nearly matches the best known polynomial algorithms by Chekuri et al. [35] and Feldman et al. [61], both of which achieve a  $O(k^{1/2+\epsilon})$ -approximation for any constant  $\epsilon > 0$ . Our bound improves upon a  $k^{1/4-o(1)}$  ratio from [52] under Gap-ETH.
- *Densest  $k$ -Subhypergraph*: given a hypergraph  $G = (V, E)$  and a positive integer  $k$ , output a  $k$ -size subset  $S \subseteq V$  that maximizes the number of hyperedges fully contained in  $S$ . The trivial algorithm that outputs any hyperedge (of size at most  $k$ ) obtains a  $2^k$ -approximation. We prove a matching lower bound (Theorem 15): no  $2^{o(k)}$ -approximation FPT (in  $k$ ) algorithm exists. This resolves an open question posed by Cygan et al. [45], assuming SPCH.



### 1.1.2 Hardness of graph pattern detection from SPCH

*Graph Pattern Detection*, also known as *Subgraph Isomorphism* and closely related to *Motif Discovery*, is a fundamental problem in graph algorithms: Given a host graph  $G$  and a pattern graph  $H$ , decide whether  $G$  contains a subgraph  $S$  isomorphic to  $H$ . There are two main variants of this problem, where  $S$  is either required to be *induced* or *not-necessarily-induced*. For both variants, there is a brute-force  $n^{O(k)}$ -time algorithm. We prove matching SPCH-hardness for both variants. In addition to beating the ETH-based state-of-the-art for these results, we highlight that our reductions to graph pattern detection problems are extremely simple (in contrast to the prior work).

For induced subgraph detection, we prove the following:

► **Theorem 3.** *Assuming the Strongish Planted Clique Hypothesis (Hypothesis 1), for every  $k$ -node pattern  $H$ , there is no algorithm that solves the induced pattern detection problem on  $n$ -vertex graphs in time  $f(k) \cdot n^{o(k)}$  for any function  $f$ .*

Our  $n^{\Omega(k)}$  lower bound for all patterns can be compared to recent work of [48], who proved: (i)  $n^{\Omega(\log(k))}$  lower bound for every pattern assuming ETH; (ii)  $n^{\Omega(\sqrt{k})}$  lower bound for every pattern assuming ETH and the Hadwiger conjecture; and (iii)  $n^{\Omega(k/\log(k))}$  lower bound for most patterns.<sup>2</sup>

For not-necessarily-induced subgraph detection, it is no longer true that every pattern is hard (e.g. it is trivial to find a not-necessarily-induced subgraph isomorphic to an independent set). But we prove (Corollary 10) that for most patterns<sup>3</sup> detection requires  $n^{\Omega(k)}$  time assuming SPCH. For comparison, [48] proved that under ETH, not-necessarily-induced subgraph detection requires  $n^{\Omega(\omega(H))}$  time, where  $\omega(H)$  denotes the clique number of the pattern  $H$ . (Note that  $\omega(H) = \Theta(\log k)$  for most patterns.)

**$k$ -biclique detection.** For the special case where the pattern is a  $k$ -biclique, our aforementioned  $n^{\Omega(k)}$  hardness for non-induced subgraph detection rules out even constant factor approximations (Corollary 9). This improves over  $n^{\Omega(\sqrt{k})}$  lower bounds under ETH for the exact case [80] or Gap-ETH for approximation [31].

**Densest  $k$ -Subgraph.** We obtain our pattern detection result by first showing a  $n^{\Omega(k)}$  running time bound for  $O(1)$ -approximating Densest  $k$ -Subgraph (Theorem 8). This improves upon the previous lower bound who give a running time lower bound of  $n^{\Omega(\log k)}$  assuming Gap-ETH [31]. The aforementioned lower bounds for pattern detection follow almost trivially from our running time lower bound for Densest  $k$ -Subgraph; see Sections 3.2 and 3.3 for more detail.

## 1.2 Techniques

The starting point for all of our reductions is a *randomized graph product*: starting with an instance  $G$  of the planted clique problem on  $n$  vertices and any integers  $\ell \leq n$  and  $N$ , we produce a graph  $G'$  by taking its vertices to be  $N$  randomly sampled subsets  $S_1, \dots, S_N$  of  $\ell$  vertices each, and we add an edge on  $S_i, S_j$  if and only if their union induces a clique in  $G$ .

<sup>2</sup> In the sense of a pattern sampled randomly from  $\mathcal{G}(k, 1/2)$ .

<sup>3</sup> In particular any pattern with a constant fraction of the  $\binom{k}{2}$  possible edges.



The randomized graph product [16] (and its derandomized variant [4]) has a long history in proving hardness of approximating Maximum Clique. While not stated explicitly, it was also used to prove parameterized inapproximability of Densest  $k$ -Subgraph in [31]. As we will explain in more detail below, the main difference between our proof and previous works lie in the soundness, where we appeal to the fact that  $G \sim \mathcal{G}(n, 1/2)$  to achieve tighter bounds.

Since we would like to prove hardness of approximating Densest  $k$ -Subgraph in the perfect completeness case, our goal is to show (for appropriately chosen values of  $N, \ell$ ) that

- (Completeness) if  $G$  contains a large clique, then with high probability so does  $G'$ , and,
- (Soundness) if  $G$  is random, then  $G'$  does not have small dense subgraphs (with high probability).

For the completeness, if the starting graph  $G$  has a  $\kappa$ -clique, then the set of  $S_i$  that fall entirely within the  $\kappa$ -clique will form a clique in  $G'$  (the expected size is  $(\frac{\kappa}{n})^\ell \cdot N$ ). This part of the proof is exactly the same as that in the aforementioned previous works.

To prove soundness, we calculate the probability that a specific  $\gamma k$ -dense  $k$ -subgraph appears in  $G'$ , then take a union bound over all  $\leq \binom{N}{k} \cdot 2^{\binom{k}{2}}$  possible such subgraphs. Our simple argument hinges on showing that a  $k$ -subgraph with  $\gamma k^2$  edges in  $G'$  induces (with high probability) at least  $\Omega(\gamma k^2 \ell^2)$  edges in  $G$ , and since any such set of edges appears in  $G$  with probability at most  $2^{-\Omega(\gamma k^2 \ell^2)}$ , by choosing  $\ell$  sufficiently large we can beat this union bound. To argue that small subgraphs with  $m$  edges in  $G'$  induce small subgraphs with  $\Omega(m \ell^2)$  edges in  $G$ , we use that the randomly chosen  $S_i$  (for an appropriate choice of  $N \ll n^\ell$  and  $k_*$  sufficiently small) form a *dispenser*: the union of *any*  $t \leq k_*$  of the  $S_i$  contains  $\Omega(t\ell)$  vertices of  $G$  with high probability.<sup>4</sup> This implies that for  $k \leq k_*$ , each  $k$ -subgraph of  $G'$  corresponds to a union of  $k$  *pairwise* mostly-non-overlapping subsets of  $\ell$  vertices. Now, since each edge between mostly non-overlapping sets in  $G'$  corresponds to a  $\Omega(\ell)$ -clique in  $G$ , this in turn can be used to show that any  $k$ -subgraph of  $G'$  with density  $\gamma k$  corresponds to a subgraph of  $G$  with  $\Omega(\gamma k^2 \ell^2)$  edges. In this way we rule out the existence of  $\gamma k$ -dense  $k$ -subgraphs in  $G'$  (with high probability).

By carefully choosing the parameters  $N, \ell, \gamma$ , to control the completeness, soundness, and reduction size, we get a fine-grained reduction from Planted  $\kappa$ -Clique to Densest  $k$ -Subgraph. Our results for other problems are obtained via direct reductions from the Densest  $k$ -Subgraph problem.

We end by stressing that our new soundness proof gives a strong quantitative improvement over prior results, which is what enables us to achieve  $k^{\Omega(1)}$  inapproximability. Specifically, the previous soundness proof from [31] – in turn adapted from [84] – relies on showing that the graph is  $t$ -biclique-free for some  $t \in \mathbb{N}$ ; they then apply the so-called Kovari-Sos-Turan theorem [77] to deduce that any  $k$ -subgraph contains at most  $O(k^{2-1/t})$  edges. Notice that this gap is only  $O(k^{1/t})$ , and  $t$  cannot be a constant as otherwise the completeness and soundness case can be distinguished in time  $n^{O(t)} = \text{poly}(n)$ . As a result, their technique cannot yield an  $k^{\Omega(1)}$ -factor inapproximability for Densest  $k$ -Subgraph. Similarly, the techniques from [31] cannot give a running time lower bound of the form  $n^{\omega(\log k)}$  for  $O(1)$ -approximation of Densest  $k$ -Subgraph. The reason is that, to get a constant gap bounded from one, they need to select  $t = O(\log k)$ . Once again, this is in contrast to our technique which yields a tight running time lower bound of  $n^{\Omega(k)}$  in this setting, although our proof requires a different starting hardness result (from SPCH).

<sup>4</sup> The fact that the randomized graph product yields a dispenser has been used in previous works as well, see e.g. [16, 98, 31].

### 1.3 Discussion and Open Questions

In this work, we have proposed the Strongish Planted Clique Hypothesis, and used it prove several tight hardness of approximation or running time lower bound results. One direction for future investigation is to use SPCH to derive other interesting lower bounds.

Another intriguing question directly related to our hardness of approximation results is whether we can strengthen the inapproximability factors from  $k^{\Omega(1)}$  to  $n^{\Omega(1)}$ . Whether it is hard to approximate Densest  $k$ -Subgraph to within a  $n^{\Omega(1)}$  factor is a well-known open problem and is related to some other conjectures, such as the Sliding-Scaling Conjecture [14]<sup>5</sup>. As such, it would be interesting if one can prove this hardness under some plausible assumption. We remark that an attempt has been made in this direction, using the so-called “Log-Density Threshold” approach [21], which posits a heuristic for predicting which average-case Densest- $k$ -Subgraph problems are hard. The approach has also been applied to other related questions [39, 40, 42]. Nonetheless, there is still little evidence that these average-case DkS problems are indeed hard; not even lower bounds against strong SDP relaxations are known, although there are some matching lower bounds against LP hierarchies [22, 41].

Finally, it is of course interesting to either refute or find more evidence supporting the Strongish Planted Clique Hypothesis. As stated earlier, the current best supporting evidence is the Sum-of-Squares lower bound from [11]. Can such a lower bound be extended to, e.g., rule out any semi-definite programs of size  $n^{o(\log n)}$  (à la [79] for CSPs)?

### 1.4 Other Related Works

Historically, postulating hardness for average-case problems has been helpful in illuminating the landscape for hardness of approximation, beginning with Feige’s seminal random-3-SAT hypothesis [57] and its numerous consequences (e.g. [50]). See also [49, 2, 7, 12].

As discussed briefly above, the Planted Clique Hypothesis (PCH), which states that there is no polynomial-time algorithm for planted clique, has many known consequences for hardness of approximation. We draw attention in particular to the work of [3], which also obtains hardness of approximation results based on PCH. But even assuming the SPCH, their results can only rule out  $n^{\text{polylog}(\kappa)}$ -time algorithms for  $2^{\log(\kappa)^{2/3}}$ -approximating densest- $\kappa$ -subgraph for  $\kappa = n^{\Omega(1)}$ . Their reduction also uses a graph product, but the set of vertices of their new graph  $G'$  contains all  $\ell$ -size subsets of vertices of  $G$ . In contrast, we employ the *randomized* graph product, where we only randomly pick some  $\ell$ -size subsets – this allows us to better control the instance size blowup, which turns out to be crucial for obtaining our tight inapproximability and running time results.

Below we discuss in more detail the previous works for each of the problems studied here.

**Densest  $k$ -Subgraph.** The problem is well-studied in the approximation algorithms literature (e.g. [58, 76, 21]). The best known polynomial time algorithm [21] achieves an approximation ratio of  $O(n^{1/4+\epsilon})$  for any  $\epsilon > 0$ . Even though the NP-hardness of Densest  $k$ -Subgraph follows immediately from that of  $k$ -Clique [74], no NP-hardness of approximation of Densest  $k$ -Subgraph even for a small factor of 1.001 is known. Nonetheless, several hardness of approximation results are known under stronger assumptions [57, 75, 90, 3, 12, 25, 84].

<sup>5</sup> Specifically, from a reduction of [34],  $n^{\Omega(1)}$ -factor hardness of approximation of Densest  $k$ -Subgraph also implies that of the Label Cover problem.

Among these, only [3, 12] and [84] yield super-constant inapproximability ratios. Specifically, [3] rules out  $2^{O((\log n)^{2/3})}$ -approximation in polynomial time under a hypothesis similar to SPCH, [84] rules out  $n^{o(1/\text{poly log log } n)}$ -approximation under ETH, and [12] rules out  $n^{O(1)}$  approximations under a strong conjecture regarding the optimality of semidefinite programs for solving every random CSP.

Our hardness result holds even in the so-called *perfect completeness* case, where we are promised that the graph  $G$  contains a  $k$ -clique. In this case, a quasi-polynomial time approximation scheme exists [60]. Braverman et al. [25] showed that this is tight: there exists a constant  $\epsilon > 0$  for which  $(1 + \epsilon)$ -approximation of Densest  $k$ -Subgraph in the perfect completeness case requires  $n^{\Omega(\log n)}$ -time assuming ETH. We remark that the hardness from [84] also applies in the perfect completeness case, but it only rules out polynomial time algorithms.

For the parameterized version of the problem, its W[1]-hardness follows immediately from that of  $k$ -Clique [54]. Chalermsook et al. [31] showed that no  $k^{o(1)}$ -approximation is achievable in FPT time, unless Gap-ETH is false. This hardness also holds in the perfect completeness case, and yields a running time lower bound of  $n^{\Omega(\log k)}$  for any constant factor approximation.

**$k$ -Biclique.** Similar to Densest  $k$ -Subgraph, while the NP-hardness for the exact version of  $k$ -Biclique has long been known (e.g. [72]), even 1.001-factor NP-hardness of approximation has not yet been proved although inapproximability results under stronger assumptions are known [57, 75, 19, 85]. Specifically, under strengthened variants of the Unique Games Conjecture, the problem is hard to approximate to within a factor of  $n^{1-\epsilon}$  for any  $\epsilon > 0$  [19, 85].

On the parameterized complexity front, whether  $k$ -Biclique is FPT (in  $k$ ) was a well-known open question (see e.g. [55]). This was resolved by Lin [80] who showed that the problem is W[1]-hard and further provided a running time lower bound of  $n^{\Omega(\sqrt{k})}$  under ETH. As stated above, this running time lower bound was extended to rule out any constant factor approximation in [31] under Gap-ETH. Furthermore, [31] showed that no  $o(k)$ -approximation exists in FPT time.

**Smallest  $k$ -Edge Subgraph.** Most of the hardness of approximation for Densest  $k$ -Subgraph easily translates to Smallest  $k$ -Edge Subgraph as well, with a polynomial loss in the factor of approximation. For example, the hardness from [84] implies that Smallest  $k$ -Edge Subgraph cannot be approximated to within a factor of  $n^{1/\text{poly log log } n}$  in polynomial time, assuming ETH. On the other hand, Chlamtac et al. [39] devised an  $O(n^{3-2\sqrt{2}+\epsilon})$ -approximation algorithm for any constant  $\epsilon > 0$  for the problem; this remains the best known approximation algorithm for the problem.

**Densest  $k$ -Subhypergraph.** Apart from the hardness results inherited from Densest  $k$ -Subgraph, not much is known about Densest  $k$ -Subhypergraph. Specifically, the only new hardness is that of Applebaum [6], who showed that the problem is hard to approximate to within  $n^\epsilon$  for some constant  $\epsilon > 0$ , assuming a certain cryptographic assumption; this holds even when each hyperedge has a constant size. On the other hand, the only (non-trivial) approximation algorithm is that of Chlamtac et al. [37] which achieves  $O(n^{0.698})$ -approximation when the hypergraph is 3-uniform.

**Steiner  $k$ -Forest.** The Steiner  $k$ -Forest problem is a generalization of several well-known problems, including the Steiner Tree problem and the  $k$ -Minimum Spanning Tree problem. This problem was first explicitly defined in [67] and subsequently studied in [93, 65, 51].

In terms of the number of vertices  $n$  of the input graph, the best known approximation ratio achievable in polynomial time is  $O(\sqrt{n})$  [65] (assuming that  $k \leq \text{poly}(n)$ ); furthermore, when edge weights are uniform, a better approximation ratio of  $O(n^{0.449})$  is achievable in polynomial time [51]. On the other hand, as stated earlier, in terms of  $k$ , the best known approximation ratio is  $O(\sqrt{k})$  [65].

A reduction in [67] together with W[1]-hardness of  $k$ -Clique [54] implies that Steiner  $k$ -Forest is W[1]-hard with respect to the parameter  $k$ . We are not aware of any further parameterized complexity study of this problem (with respect to parameter  $k$ ).

**Directed Steiner Network.** Several polynomial time approximation algorithms have been proposed for the Directed Steiner Network problem [33, 35, 61, 15, 38, 1]; in terms of the number of vertices  $n$  of the input graph the best known approximation ratio is  $O(n^{2/3+\epsilon})$  [15] and in terms of  $k$  the best known ratio is  $O(k^{1/2+\epsilon})$  for any constant  $\epsilon > 0$  [35, 61]. On the hardness front, Dodis and Khanna [53] shows that the problem is quasi-NP-hard to approximate to within a factor of  $2^{(\log n)^{1-\epsilon}}$  for any constant  $\epsilon > 0$ . Furthermore, Guo et al. [64] show that the exact version of the problem is W[1]-hard with respect to parameter  $k$ . Later, [52] rules out even  $k^{1/4-o(1)}$ -approximation in FPT time, under Gap-ETH.

**Graph Pattern Detection.** As discussed earlier, [48] give ETH-based hardness results for graph pattern detection, both in the induced and non-induced case. The complexity for many special patterns has also been considered, e.g.  $k$ -cliques,  $k$ -bicliques (mentioned above), and  $k$ -cycles (e.g. [5, 97, 47, 48, 81]). A  $k$ -clique can be detected in time  $O(n^{\lceil k/3 \rceil \omega})$  using fast matrix multiplication [89], and the  $k$ -Clique Conjecture in Fine-Grained Complexity postulates that this is essentially optimal [95]. Any other pattern over  $k$  vertices can be detected in time  $O(n^{k-1})$ , without using fast matrix multiplication [24]. There is also an extensive body of work on *counting* the number of occurrences of a pattern in a host graph (e.g. [88, 78, 94, 96, 44, 43]).

## Preliminaries and Notation

For a natural number  $n \in \mathbb{N}$ , we use  $[n]$  to denote the set of integers up to  $n$ ,  $[n] = \{1, \dots, n\}$ . We will use the abbreviation “w.h.p.” to mean “with high probability.”

For an undirected graph  $G = (V, E)$ , we use  $\deg_G(v)$  to denote the degree of a vertex  $v \in V$ , and  $\min\text{-deg}(G)$  to denote  $\min_{v \in V(G)} \deg_G(v)$ . For a subset  $S \subseteq V$ , we use  $G[S] = (V, E[S])$  to denote the induced subgraph of  $G$  on subset  $S$ .

The *density* of  $G$ , denoted by  $\text{den}(G)$ , is defined as  $|E|/|V|$ . We use  $\text{den}_{\leq k}(G)$  to denote  $\max_{S \subseteq V, |S| \leq k} \text{den}(G[S])$ , the maximum density of subgraphs of  $G$  of at most  $k$  vertices.

## 2 Randomized Graph Product

In this section we formally define our reduction, and analyze its soundness and completeness in terms of the reduction parameters (in later sections we instantiate the parameters differently for each target bound). Our reduction takes as input a graph and applies the *randomized graph product* [16], described in Figure 1. We use  $\text{RP}_{N,\ell}(G)$  to denote the distribution of outputs of the above reduction on input graph  $G$ .

We will show that for well-chosen  $N$  and  $\ell$ , the randomized graph product  $\text{RP}_{N,\ell}$  reduces the planted  $n^\delta$ -clique problem to densest- $k$  subgraph for  $k = k(N, \ell, \delta, n)$ . That is, a sample from  $\text{RP}_{N,\ell} \circ \mathcal{G}(n, \frac{1}{2})$  has no dense  $k$ -subgraph with probability close to 1, and if on the other hand  $G$  is a graph with an  $n^\delta$ -clique, then a sample from  $\text{RP}_{N,\ell}(G)$  has a dense  $k$ -subgraphs with probability close to 1.

**Input:**  $n$ -vertex Graph  $G = (V, E)$ , positive integers  $N, \ell$ .  
**Output:** Graph  $G' = (V', E')$ .  
The graph  $G'$  is constructed as follows.

1. For each  $i \in [N]$ , sample  $S_i \subseteq V$  by independently sampling  $\ell$  vertices uniformly from  $V$ .
2. Let  $V' = \{S_1, \dots, S_N\}$ .
3. For every distinct  $i, j \in [N]$ , include  $(S_i, S_j)$  in  $E'$  iff  $S_i \cup S_j$  induces a clique in  $G$ .

■ **Figure 1** Randomized Graph Product [16].

## 2.1 Completeness

We first prove that applying the randomized graph product to a graph with a large clique results in a graph with a large clique.

► **Lemma 4 (Completeness).** *Suppose that  $\delta \in (0, 1)$ ,  $N, \ell, k \in \mathbb{N}$  are such that  $N \geq 10k \cdot n^{(1-\delta)\ell}$  and  $k \geq 20$ . If  $G$  contains a clique of size  $\lceil n^\delta \rceil$ , then*

$$\Pr_{G' \sim \text{RP}_{N, \ell}(G)} [G' \text{ contains a } k\text{-clique}] \geq 0.9.$$

**Proof.** Let  $C \subseteq V$  be the  $\lceil n^\delta \rceil$ -size clique in  $G$ . For each  $i \in [N]$ ,  $\Pr[S_i \subseteq C] = \left(\frac{\lceil n^\delta \rceil}{n}\right)^\ell \geq n^{-(1-\delta)\ell}$ . By our lower bound on  $N$ , the expected number of indices  $i \in [N]$  such that  $S_i \subseteq C$  is at least  $10k$ , and thus a Chernoff bound implies that with probability  $1 - \exp(-4k) \geq 0.9$ , there exists at least  $k$  indices  $i \in [N]$  such that  $S_i \subseteq C$ . By definition of the randomized graph product  $\text{RP}_{N, \ell}$ , these subsets form a clique in  $G'$ . ◀

## 2.2 Soundness

We now prove that if we apply the randomized graph product to a graph drawn from  $\mathcal{G}(n, \frac{1}{2})$ , with high probability the resulting graph has no small subgraphs which are too dense.

► **Lemma 5 (Soundness).** *Suppose that  $\delta \in (0, 1)$ ,  $N, \ell, k \in \mathbb{N}$  are such that  $N \leq 1000k \cdot n^{(1-\delta)\ell}$ ,  $\ell \geq k \geq 20$  and  $n^{0.99\delta} \geq k\ell$ . If  $G$  is drawn from  $\mathcal{G}(n, \frac{1}{2})$ , then*

$$\Pr_{\substack{G \sim \mathcal{G}(n, \frac{1}{2}) \\ G' \sim \text{RP}_{N, \ell}(G)}} \left[ \text{den}_{\leq k}(G') \leq \frac{10^7 \log n}{\ell \delta^2} \right] \geq 0.9.$$

We will use the following observation that allows us to translate a large-density graph to a subgraph with large minimum degree. This observation is folklore and appears e.g. in [32].

► **Observation 6.** *For any  $H = (V_H, G_H)$ , there exists  $S' \subseteq V_H$  such that  $\min\text{-deg}(H[S']) \geq \text{den}(H)$ .*

Another auxiliary lemma that is useful for us is that for any subset  $M \subseteq [N]$  not too large, the size of the union  $|\cup_{j \in M} S_j|$  is not too small relative to  $|M| \cdot \ell$ . This lemma is also standard and has been used in prior works (e.g. [16, 98, 31]).

► **Lemma 7.** *Suppose  $N \leq 1000\ell n^{(1-\delta)\ell}$ ,  $20 \leq \ell$ . Then with probability at least 0.95 over a sample  $G' \sim \text{RP}_{N, \ell} \circ \mathcal{G}(n, \frac{1}{2})$ ,  $G' = (\{S_i\}_{i \in [N]}, E')$ , the following event occurs: for every  $M \subseteq [N]$  with  $|M| \leq n^{0.99\delta}/\ell$ , we have  $|\cup_{i \in M} S_i| \geq 0.01\delta|M|\ell$ .*

## 10:10 The Strongish Planted Clique Hypothesis and Its Consequences

The proofs of both Observation 6 and Lemma 7 are included in the full version. We now prove the soundness guarantee.

**Proof of Lemma 5.** We will assume that the event in Lemma 7 occurs and show that conditioned on this event, under the randomness of  $G$ , with probability 0.95 all  $k$ -subgraphs of  $G'$  have density at most  $d := 10^7 \log n / (\ell \delta^2)$ . Lemma 5 then follows immediately since  $(0.95)^2 > 0.9$ .

Consider any  $J \subseteq [N]$  of size  $k' \leq k$ . For brevity, let  $\mathcal{F}(J)$  denote the set of all graphs whose vertices are  $S_j$  for  $j \in J$  and the minimum degree is at least  $d$ . For each  $F = (\{S_j\}_{j \in J}, E_F) \in \mathcal{F}(J)$ , let  $\mathcal{E}^G(F)$  denote  $\bigcup_{\{S_j, S_{j'}\} \in E_F} \{\{u, v\} \mid u \in S_j, v \in S_{j'} \text{ and } u \neq v\}$ , or in words, the set of edges of  $G$  which have one endpoint in  $S_j$  and one endpoint in  $S_{j'}$  for an edge  $(S_j, S_{j'}) \in E_F$ . Observe that

$$\Pr_G[G'[\{S_j\}_{j \in J}] = F] \leq \Pr_G[\mathcal{E}^G(F) \subseteq E] = 2^{-|\mathcal{E}^G(F)|}, \quad (1)$$

where the inequality follows by definition of the randomized graph product – since  $(S_i, S_j)$  is an edge if and only if  $S_i \cup S_j$  is a clique in  $G$ , the event  $G'[\{S_j\}_{j \in J}] = F$  contains the event  $\mathcal{E}^G(F) \subseteq E$  – and the final equality follows because  $G \sim \mathcal{G}(n, \frac{1}{2})$ . To bound  $|\mathcal{E}^G(F)|$ , let  $\mathcal{V}^G(F) := \bigcup_{j \in J} S_j$ .

Since we have conditioned on the event in Lemma 7 occurring,<sup>6</sup> we have  $|\mathcal{V}^G(F)| \geq 0.01\delta k' \ell$ . Now, consider any  $v \in \mathcal{V}^G(F)$ ; from definition of  $\mathcal{V}^G(F)$ ,  $v \in S_j$  for some  $j \in J$ . Since  $F \in \mathcal{F}(J)$ ,  $S_j$  must have at least  $d$  neighbors in  $F$ . Let  $S_{j_1}, \dots, S_{j_{d'}}$  denote  $S_j$ 's neighbors in  $F$ , with  $d' \geq d$ . By applying the bound in Lemma 7, we have  $|S_{j_1} \cup \dots \cup S_{j_{d'}}| \geq 0.01\delta d' \ell \geq 0.01\delta d \ell$ . In other words,  $v$  has degree at least  $0.01\delta d \ell - 1 \geq 0.005\delta d \ell$  in the graph  $(\mathcal{V}^G(F), \mathcal{E}^G(F))$ . This implies that

$$|\mathcal{E}^G(F)| \geq \frac{1}{2} (0.01\delta k' \ell) (0.005\delta d \ell) \geq 10^{-5} \delta^2 k' d \ell^2.$$

Plugging the above bound back into (1), we have

$$\Pr_G[G'[\{S_j\}_{j \in J}] = F] \leq 2^{-10^{-5} \delta^2 k' d \ell^2} \quad (2)$$

We can use the above inequality to bound the probability that  $\{S_j\}_{j \in J}$  induces a subgraph with minimum degree at least  $d$  as follows:

$$\begin{aligned} \Pr[\min\text{-deg}(G'[\{S_j\}_{j \in J}]) \geq d] &= \sum_{F \in \mathcal{F}(J)} \Pr_G[G'[\{S_j\}_{j \in J}] = F] \stackrel{(2)}{\leq} 2^{(k')^2} \cdot 2^{-10^{-5} \delta^2 k' d \ell^2} \\ &\leq 2^{-10^{-6} \delta^2 k' d \ell^2}. \end{aligned}$$

where the first inequality follows because there are at most  $2^{(k')^2}$  subgraphs of an  $k'$ -vertex graph, and to obtain the final inequality we have applied that  $\ell \geq k \geq k'$  and  $d \geq 10^7 / (\delta^2 \ell)$ .

Applying Observation 6, the existence of a  $k$ -subgraph with density at least  $d$  would imply the existence of some  $J \subseteq [N]$  with  $|J| \leq k$  and minimum degree at least  $d$ . Taking union bound over all  $J \subseteq [N]$  of size at most  $k$  and applying our above bound, we have

$$\Pr[\text{den}_{\leq k}(G)] \leq \sum_{k'=1}^k N^{k'} \cdot 2^{-10^{-6} \delta^2 k' d \ell^2} = \sum_{k'=1}^k \left( N \cdot 2^{-10^{-6} \delta^2 d \ell^2} \right)^{k'}$$

<sup>6</sup> Note that  $|J| = k' \leq k \leq n^{0.99\delta} / \ell$  by our assumption and hence  $J$  satisfies the condition in Lemma 7.



$$\begin{aligned}
&\leq \sum_{k'=1}^k \left( 8 \cdot n \cdot 2^{-10^{-6}\delta^2 d \ell} \right)^{k' \ell} \\
&\leq \sum_{k'=1}^k 0.01^{k' \ell} \leq 0.95,
\end{aligned}$$

where to obtain the second line we use that  $N \leq 1000kn^{(1-\delta)\ell}$  and that  $\ell \geq k \geq 20$ , and in the final line we use that  $d \geq 10^7 \log n / (\ell\delta)^2$ . This completes our proof.  $\blacktriangleleft$

### 3 Tight Running Time Lower Bounds

In this section, we prove our tight running time lower bounds for  $O(1)$ -approximating Densest  $k$ -Subgraph,  $O(1)$ -approximating  $k$ -Biclique and (exact) Graph Pattern Detection.

#### 3.1 Constant Approximation for Densest $k$ -Subgraph

We start with the  $n^{\Omega(k)}$  running time lower bound for  $O(1)$ -approximating Densest  $k$ -Subgraph, from which the remaining results easily follow. We remark that this running time lower bound improves upon that of  $n^{\Omega(\log k)}$ , which is implicit in [31].

**► Theorem 8.** *Assuming Hypothesis 1, for any constant  $C > 0$ , there is no  $f(k) \cdot n^{o(k)}$ -time algorithm that can approximate Densest  $k$ -Subgraph to within a factor  $C$  for any function  $f$ . Furthermore, this holds even in the perfect completeness case where the input graph is promised to contain a  $k$ -clique.*

We will prove Theorem 8 by simply selecting an appropriate setting of parameters for the randomized graph product. Specifically, we will let  $\ell = O(C \log n / k)$  and  $N = n^{O(\ell)} = n^{o(\log n)}$ ; the generic soundness lemma (Lemma 5) then implies that the density of any  $k$ -subgraph in the soundness case is at most  $O(k/C)$  which yields the desired  $C$  inapproximability factor.

**Proof of Theorem 8.** We will reduce the problem of distinguishing samples from  $\mathcal{G}(n, \frac{1}{2})$  vs.  $\mathcal{G}(n, \frac{1}{2}, \lceil n^\delta \rceil)$  to approximating Densest  $k$ -subgraph.

For  $C$  the constant specified in the statement of the theorem, choose  $\ell = \lceil \frac{10^8 C \log n}{\delta^2 k} \rceil$  and  $N = \lceil 100kn^{(1-\delta)\ell} \rceil$ , so that  $d = \frac{10^7 \log n}{\ell \delta^2} \leq \frac{k}{10C}$ . Given a graph  $G$ , we sample  $G' \sim \text{RP}_{N,\ell}(G)$ .

**Completeness:** If  $G \sim \mathcal{G}(n, \frac{1}{2}, \lceil n^\delta \rceil)$ , then by Lemma 4 and since  $N \geq 10kn^{(1-\delta)\ell}$ , we have that with probability at least  $1 - \exp(-4k) \geq 0.9$ ,  $G' = \text{RP}_{N,\ell}(G)$  has a  $k$ -clique, and so  $\text{den}_{\leq k}(G') \geq k - 1$ .

**Soundness:** For any  $20 \leq k \leq \ell$  and any  $\delta$  bounded away from 0, we clearly satisfy the requirement of Lemma 5 that  $k\ell \leq n^{0.99\delta}$  and that  $N \leq 1000kn^{(1-\delta)\ell}$  for any sufficiently large  $n$ . Hence, if  $G \sim \mathcal{G}(n, \frac{1}{2})$ , applying the Lemma to  $G' \sim \text{RP}_{N,\ell}(G)$  we have that with probability at least 0.9,  $\text{den}_{\leq k}(G') \leq d \leq \frac{k}{10C}$ .

Thus, any algorithm which approximates Densest  $k$ -subgraph up to a factor of  $C$  in time  $f(k)N^{\epsilon k}$  can solve the planted  $\lceil n^\delta \rceil$ -clique problem in time  $f(k)(100kn^{(1-\delta)\ell})^{\epsilon k} = g(k)n^{\frac{(1-\delta)\delta^2 C}{10^8} \epsilon \log n}$  for  $g(k) = f(k)(100k)^{\epsilon k}$ . This contradicts Hypothesis 1 whenever  $\lim_{n \rightarrow \infty} \epsilon = 0$ . Choosing  $k$  to be a sufficiently slowly growing function of  $n$ , for any  $\epsilon$  decreasing in  $k$  we have a contradiction of the Hypothesis. This concludes the proof.  $\blacktriangleleft$

### 3.2 $O(1)$ -Approximation for $k$ -Biclique

Recall that a  $k$ -biclique, denoted by  $K_{k,k}$ , is a complete bipartite graph where each side has exactly  $k$  vertices. In the  $k$ -Biclique problem, we are given an undirected graph  $G$  and a positive integer  $k$ . Further, we are promised that  $G$  contains a  $K_{k,k}$  as a subgraph. Our goal is to output a balanced biclique in  $G$  of size as large as possible. (Note that we say that an algorithm achieves  $\alpha$ -approximation if the output biclique has size at least  $k/\alpha$ .)

We prove the following tight running time lower bound for  $O(1)$ -approximation of  $k$ -Biclique:

► **Corollary 9.** *Assuming Hypothesis 1, for any constant  $C > 0$ , there is no  $f(k) \cdot n^{o(k)}$ -time algorithm that can approximate  $k$ -Biclique to within a factor  $C$  for any function  $f$ . Furthermore, this holds even when we are promised that the input graph contains a  $2k$ -clique.*

**Proof.** Suppose contrapositively that there is an  $f(k) \cdot n^{o(k)}$ -time algorithm  $\mathbb{A}$  that can approximate  $k$ -Biclique to within a factor of  $C$  for some function  $f$ . We may use it to approximate Densest  $k$ -Subgraph with perfect completeness as follows. On a given instance  $(G, k)$  of Densest  $k$ -Subgraph with perfect completeness, we run our algorithm  $\mathbb{A}$  on  $(G, \lfloor k/2 \rfloor)$ . From the approximation guarantee of  $\mathbb{A}$ , we will find a  $t$ -Biclique for  $t \geq \lfloor k/2 \rfloor / C \geq \frac{k}{4C}$ . By taking this biclique together with arbitrary  $(k - 2t)$  additional vertices, we find a subset of size  $k$  that induces at least  $t^2 \geq \frac{k^2}{16C^2}$  edges. Hence, we have found a  $16C^2$ -approximation for Densest  $k$ -Subgraph in time  $f(k) \cdot n^{o(k)}$ , which by Theorem 8 violates Hypothesis 1. ◀

### 3.3 Pattern detection

Theorem 8 also yields the following corollary for the running time of pattern detection:

► **Corollary 10.** *Assuming Hypothesis 1, for almost all  $k$ -node patterns  $H$ , the not necessarily induced pattern detection problem cannot be solved in time  $f(k) \cdot n^{o(k)}$ .*

In the statement of Corollary 10, by “almost all  $k$ -node patterns” we mean w.h.p. over  $H \sim \mathcal{G}(k, 1/2)$ .<sup>7</sup>

**Proof.** By standard concentration inequalities, most  $H \sim \mathcal{G}(k, \frac{1}{2})$  have average degree  $\frac{k}{2} \pm o(k)$ . For such a pattern  $H$ , an algorithm that solves the not necessarily induced pattern detection problem also gives  $O(1)$ -approximation for Densest  $k$ -Subgraph. Hence, the corollary immediately follows from Theorem 8. ◀

► **Theorem (Restatement of Theorem 3).** *Assuming the Strongish Planted Clique Hypothesis (Hypothesis 1), for every  $k$ -node pattern  $H$ , there is no algorithm that solves the induced pattern detection problem on  $n$ -vertex graphs in time  $f(k) \cdot n^{o(k)}$  for any function  $f$ .*

**Proof.** We assume that  $H$  is at least  $\frac{k}{4}$ -dense (that is, has average degree at least  $\frac{k}{2}$ ). This is without loss of generality as otherwise, we may take the complement of  $H$  – for induced pattern detection the complexity is the same.

We start our reduction from an instance  $G$  of Densest  $k$ -Subgraph as in Theorem 8. We randomly color all the vertices of  $G$  in  $k$  colors, one for each vertex of  $H$ . We construct a graph  $G'$  from  $G$  by keeping edge  $(u, v) \in G$  if and only if  $u, v$  have different colors, and the vertices in  $H$  corresponding to those colors share an edge. (Note that we do not add any edges.)

<sup>7</sup> It may be more natural to sample uniformly from all *unlabeled*  $k$ -node patterns, but w.h.p. a graph drawn from  $\mathcal{G}(k, 1/2)$  has no non-trivial automorphisms [56], so the two notions of “almost all  $k$ -node patterns” are in fact equivalent.



**Completeness:** If  $G$  has a  $k$ -clique, with probability at least  $k^{-k}$  it has  $k$  different colors. In this case, the same vertices in  $G'$  will form an induced copy of  $H$ .

**Soundness:** If  $G$  does not have a  $\frac{k}{4}$ -dense  $k$ -subgraph, this remains true after removing edges.

Hence  $G'$  also does not contain any  $\frac{k}{4}$ -dense  $k$ -subgraph – and in particular no copy of  $H$ . As a result, if we can solve the induced pattern detection problem in time  $f(k) \cdot n^{o(k)}$ , we can achieve 4-approximation for Densest  $k$ -Subgraph with probability  $k^{-k}$  in time  $f(k) \cdot n^{o(k)}$ . By repeating this construction  $100 \cdot k^k$  times, we can achieve 4-approximation for Densest  $k$ -Subgraph with probability 0.99 in time  $O(f(k) \cdot k^k) \cdot n^{o(k)}$ . Together with Theorem 8, this concludes our proof. ◀

## 4 Tight Inapproximability Results

In this section, we prove tight inapproximability results for Densest  $k$ -Subgraph, Smallest  $k$ -Edge Subgraph, Steiner  $k$ -Forest, Directed Steiner Network, and Densest  $k$ -Subhypergraph.

### 4.1 Densest $k$ -Subgraph

We start with the  $o(k)$ -factor hardness of Densest  $k$ -Subgraph (Theorem 2), from which our other results follow. The proof of Theorem 2 is once again via selecting an appropriate setting of parameters for the randomized graph product. Specifically, we will select  $\ell = O((\log n) \cdot g(k)/k)$  where  $g(k) = o(k)$  is the assumed approximation ratio and  $N = n^{O(\ell)} = n^{o(\log n)}$ ; the generic soundness lemma (Lemma 5) then implies that the density of any  $k$ -vertex subgraph in the soundness case is at most  $k/g(k)$  as desired. The arguments are formalized below.

**Proof of Theorem 2.** We will reduce the problem of distinguishing samples from  $\mathcal{G}(n, \frac{1}{2})$  vs.  $\mathcal{G}(n, \frac{1}{2}, \lceil n^\delta \rceil)$  to approximating Densest  $k$ -subgraph on an  $N$  vertex graph.

Let  $g(k) \leq k$  be any function growing with  $k$ . Choose  $\ell = \lceil \frac{10^8 g(k) \log n}{\delta^2 k} \rceil$  and  $N = \lceil 100kn^{(1-\delta)\ell} \rceil$ , so that  $d = \frac{10^7 \log n}{\ell \delta^2} \leq \frac{k}{10g(k)}$ . Given a graph  $G$ , we sample  $G' \sim \text{RP}_{N,\ell}(G)$ .

**Completeness:** If  $G \sim \mathcal{G}(n, \frac{1}{2}, \lceil n^\delta \rceil)$ , then just as in the proof of Theorem 8, Lemma 4 implies that with probability at least 0.9,  $G' = \text{RP}_{N,\ell}(G)$  has a  $k$ -clique.

**Soundness:** For any  $20 \leq k \leq \ell$  and  $\delta$  bounded away from 0, we satisfy the requirements of Lemma 5 (just as in the proof of Theorem 8). Applying the lemma, if  $G \sim \mathcal{G}(n, \frac{1}{2})$ , we conclude that with probability at least 0.9,  $\text{den}_{\leq k}(G') \leq d \leq \frac{k}{10g(k)}$ . This means that

any  $k$ -vertex subgraph of  $G'$  contains at most  $\frac{k^2}{10g(k)} < \frac{\binom{k}{2}}{g(k)}$  edges.

Hence, any algorithm which approximates Densest  $k$ -subgraph within  $g(k)$  in time  $f(k)\text{poly}(N)$  can solve the planted  $\lceil n^\delta \rceil$ -clique problem in time  $f(k) \cdot \text{poly}(100kn^{(1-\delta)\ell}) = h(k) \cdot \text{poly}\left(n^{\frac{(1-\delta)\delta^2}{10^8} \frac{g(k)}{k} \log n}\right)$  for  $h(k) = f(k)\text{poly}(100k)$ . Choosing  $k$  to be a sufficiently slowly growing function of  $n$ , for any function  $g(k)$  with  $\lim_{n \rightarrow 0} \frac{g(k)}{k} = 0$  we have a contradiction of Hypothesis 1, as desired. ◀

### 4.2 Smallest $k$ -Edge Subgraph

► **Corollary 11.** *Assuming Hypothesis 1, there is no  $f(k) \cdot \text{poly}(n)$ -time algorithm that can approximate Smallest  $k$ -Edge Subgraph to within a factor  $o(\sqrt{k})$  for any function  $f$ .*

We prove the above corollary by reducing from Densest  $k$ -Subgraph; we remark here that similar reductions between the two problems are folklore and have appeared before in literature, e.g. in [67]. Hence, we defer the proof to the full version of the paper.

### 4.3 Steiner $k$ -Forest

► **Corollary 12.** *Assuming Hypothesis 1, there is no  $f(k) \cdot \text{poly}(n)$ -time algorithm that can approximate Steiner  $k$ -Forest to within a factor  $o(\sqrt{k})$  for any function  $f$ .*

Corollary 12 immediately follows from Corollary 11 via the following reduction:

► **Theorem 13** ([67, Theorem 6.2]). *If there is an  $f(k) \cdot \text{poly}(n)$ -time  $g(k)$ -approximation algorithm for Steiner  $k$ -Forest, then there is an  $f(k) \cdot \text{poly}(n)$ -time  $g(k)$ -approximation algorithm for Smallest  $k$ -Edge Subgraph.*

### 4.4 Directed Steiner Network

► **Corollary 14.** *Assuming Hypothesis 1, there is no  $f(k) \cdot \text{poly}(n)$ -time algorithm that can approximate Directed Steiner Network with  $k$  terminal pairs to within a factor  $o(\sqrt{k})$  for any function  $f$ .*

Corollary 14 is shown via a reduction from Smallest  $p$ -Edge Subgraph, which is similar to the reduction from the Label Cover problem by Dodis and Khanna [53] that was also used in subsequent works [36, 52]. The proof is deferred to the full version.

### 4.5 Densest $k$ -Subhypergraph

► **Theorem 15.** *Assuming Hypothesis 1, there is no  $f(k) \cdot \text{poly}(n)$ -time algorithm that can approximate Densest  $k$ -Subhypergraph to within a factor  $2^{o(k)}$  for any function  $f$ .*

The proof of our inapproximability for Densest  $k$ -Subhypergraph (Theorem 15) is unlike the others in this section: instead of starting from the inapproximability of Densest  $k$ -Subgraph (Theorem 2), we will start from the tight running time lower bound for  $O(1)$ -approximate  $k$ -Biclique (Corollary 9).

#### 4.5.1 A Combinatorial Lemma

Before we describe our reduction, let us prove the following lemma, which bounds the number of (induced) copies of  $K_{\ell,\ell}$  in a  $K_{t,t}$ -free graph for  $\ell < t$ . This is a generalized setting of the classic Kovari-Sos-Turan (KST) theorem [77], which applies only to the case  $\ell = 1$ . Note that, due to the parameters of interest in our reduction, we only prove a good bound for large  $\ell$ ; for  $\ell = 1$ , the bound in our lemma is trivial (and hence weaker than the KST theorem).

► **Lemma 16.** *Let  $\kappa, t, \ell$  be positive integers such that  $\ell < t \leq \kappa/16$ . Then, for any  $\kappa$ -vertex  $K_{t,t}$ -free graph  $H$ , the number of (not necessarily induced) copies of  $K_{\ell,\ell}$  in  $H$  is at most  $\left(2 \cdot e^{-\frac{\ell^2}{16t}}\right) \cdot \binom{\kappa}{\ell} \binom{\kappa-\ell}{\ell}$ .*

**Proof.** Let  $V$  denote the vertex set of  $H$ . We will count the number of copies of  $K_{\ell,t}$  in  $H$  in two ways. First, for every subset  $S \in \binom{V}{\ell}$ , the number of  $(\ell, t)$ -bicliques of the form  $(S, T)$  where  $T \in \binom{V}{t}$  is  $\binom{|N(S)|}{t}$  where  $N(S)$  denote the set of common neighbors of  $S$ . Hence, in total the number of  $(\ell, t)$ -bicliques in  $H$  is  $\sum_{S \in \binom{V}{\ell}} \binom{|N(S)|}{t}$ . On the other hand, for every set  $T \in \binom{V}{t}$ , we must have  $|N(T)| \leq t - 1$  since  $H$  does not contain  $K_{t,t}$ . As a result, the number of  $(\ell, t)$ -bicliques of the form  $(S, T)$  for a fixed  $T$  is at most  $\binom{t-1}{\ell}$ . That is, in total, there can be at most  $\binom{\kappa}{t} \binom{t-1}{\ell}$  copies of  $K_{\ell,t}$  in the graph. This implies that

$$\binom{\kappa}{t} \binom{t-1}{\ell} \geq \sum_{S \in \binom{V}{\ell}} \binom{|N(S)|}{t}. \quad (3)$$

For brevity, let  $\lambda$  denote  $(\kappa - \ell)/t$ ,  $\gamma$  denote  $\lambda^{-\ell/(2t)}$  and let  $x$  denote  $\gamma \cdot \kappa + (1 - \gamma) \cdot t$  (note  $x$  is chosen so that  $\frac{x-t}{\kappa-t} = \gamma$ ). Let us now classify  $S \in \binom{V}{\ell}$  into two groups: those with  $|N(S)| \geq x$  and those with  $|N(S)| < x$ . That is, we define

$$\mathcal{S}_{\geq x} := \left\{ S \in \binom{V}{\ell} \mid |N(S)| \geq x \right\}, \quad \text{and} \quad \mathcal{S}_{< x} := \left\{ S \in \binom{V}{\ell} \mid |N(S)| < x \right\}.$$

From (3), we have  $\binom{\kappa}{t} \binom{t-1}{\ell} \geq \sum_{S \in \mathcal{S}_{\geq x}} \binom{|N(S)|}{t} \geq |\mathcal{S}_{\geq x}| \cdot \binom{\lceil x \rceil}{t}$ . Rearranging, we have

$$\begin{aligned} |\mathcal{S}_{\geq x}| &\leq \frac{\binom{t-1}{\ell} \binom{\kappa}{t}}{\binom{\lceil x \rceil}{t}} \leq \binom{t-1}{\ell} \left( \frac{\kappa-t}{x-t} \right)^t = \binom{t-1}{\ell} \gamma^{-t} \leq \binom{\kappa}{\ell} \left( \frac{t-1}{\kappa} \right)^\ell \gamma^{-t} \\ &\leq \binom{\kappa}{\ell} \lambda^{-\ell} \gamma^{-t} \\ &= \binom{\kappa}{\ell} \lambda^{-\ell/2} \leq \binom{\kappa}{\ell} 2^{-\ell/2}, \end{aligned} \quad (4)$$

where the last line follows because  $t, \ell \leq \frac{1}{3}\kappa$ .

Let us now count the number of  $K_{\ell, \ell}$  in  $H$ . For each fixed  $S \in \binom{V}{\ell}$ , the number of  $K_{\ell, \ell}$  of the form  $(S, T)$  where  $T \in \binom{V}{\ell}$  is exactly  $\binom{|N(S)|}{\ell}$ . Thus, the total number of  $K_{\ell, \ell}$  in  $G$  is  $\sum_{S \in \binom{V}{\ell}} \binom{|N(S)|}{\ell}$ . This term can be further bounded by

$$\begin{aligned} \sum_{S \in \binom{V}{\ell}} \binom{|N(S)|}{\ell} &= \sum_{S \in \mathcal{S}_{\geq x}} \binom{|N(S)|}{\ell} + \sum_{S \in \mathcal{S}_{< x}} \binom{|N(S)|}{\ell} \\ &\leq |\mathcal{S}_{\geq x}| \binom{\kappa - \ell}{\ell} + |\mathcal{S}_{< x}| \binom{x}{\ell} \\ &\stackrel{(4)}{\leq} 2^{-\ell/2} \binom{\kappa}{\ell} \binom{\kappa - \ell}{\ell} + \binom{\kappa}{\ell} \binom{\lfloor x \rfloor}{\ell} \\ &\leq 2^{-\ell/2} \binom{\kappa}{\ell} \binom{\kappa - \ell}{\ell} + \binom{\kappa}{\ell} \binom{\kappa - \ell}{\ell} \cdot \left( \frac{x}{\kappa - \ell} \right)^\ell \\ &\leq 2^{-\ell/2} \binom{\kappa}{\ell} \binom{\kappa - \ell}{\ell} + \binom{\kappa}{\ell} \binom{\kappa - \ell}{\ell} \cdot \left( 1 - \frac{(1 - \gamma)(\kappa - t) - \ell}{\kappa - \ell} \right)^\ell \\ &= 2^{-\ell/2} \binom{\kappa}{\ell} \binom{\kappa - \ell}{\ell} + \binom{\kappa}{\ell} \binom{\kappa - \ell}{\ell} \cdot \left( 1 - \frac{(1 - \gamma - \frac{\ell}{\kappa - t})(\kappa - t)}{\kappa - \ell} \right)^\ell \\ (\text{From } \ell < t \leq \kappa/2) &\leq 2^{-\ell/2} \binom{\kappa}{\ell} \binom{\kappa - \ell}{\ell} + \binom{\kappa}{\ell} \binom{\kappa - \ell}{\ell} \cdot (1 - 0.5(1 - \gamma - 2\ell/\kappa))^\ell \\ &\leq 2^{-\ell/2} \binom{\kappa}{\ell} \binom{\kappa - \ell}{\ell} + \binom{\kappa}{\ell} \binom{\kappa - \ell}{\ell} \cdot e^{-0.5\ell(1 - \gamma - 2\ell/\kappa)} \end{aligned} \quad (5)$$

Consider the term  $(1 - \gamma - 2\ell/\kappa)$ . We can bound it as follows:

$$\begin{aligned} (1 - \gamma - 2\ell/\kappa) &= \left( 1 - \frac{1}{\lambda^{\ell/2t}} - \frac{2\ell}{\kappa} \right) \\ (\text{From } \lambda = (\kappa - \ell)/t \geq 2) &\geq \left( 1 - 0.5^{\ell/2t} - \frac{2\ell}{\kappa} \right) \\ (\text{Bernoulli inequality}) &\geq \left( 1 - \left( 1 - \frac{\ell}{4t} \right) - \frac{2\ell}{\kappa} \right) = \left( \frac{\ell}{4t} - \frac{2\ell}{\kappa} \right) \geq \frac{\ell}{8t}. \end{aligned}$$

Plugging this back into (5), we have

$$\sum_{S \in \binom{A}{\ell}} \binom{|N(S)|}{\ell} \leq \binom{\kappa}{\ell} \binom{\kappa - \ell}{\ell} \cdot \left(2^{-\ell/2} + e^{-\frac{\ell^2}{16t}}\right) \stackrel{(\text{From } \ell < t)}{\leq} \binom{\kappa}{\ell} \binom{\kappa - \ell}{\ell} \cdot \left(2 \cdot e^{-\frac{\ell^2}{16t}}\right),$$

which concludes our proof.  $\blacktriangleleft$

#### 4.5.2 Proof of Theorem 15

**Proof of Theorem 15.** Suppose contrapositively that there is an  $f(\rho) \cdot \text{poly}(n)$ -time  $2^{\rho/g(\rho)}$ -approximation algorithm  $\mathbb{A}$  for Densest  $\rho$ -Subhypergraph where  $g = \omega(1)$ . We will construct an algorithm  $\mathbb{B}$  that achieves  $O(1)$ -approximation for  $k$ -Biclique with the promise that a  $2k$ -Clique exists in time  $h(k) \cdot n^{o(k)}$  for some function  $h$ . Theorem 15 then follows from Corollary 9.

We define  $\mathbb{B}$  on input  $(G = (V, E), k)$  as follows:

- Let  $\rho = 2k$  and  $\ell = \lceil \rho/g(\rho)^{0.1} \rceil$ .
- Construct a hypergraph  $G'$  with the same vertex set as  $G$ , and we add a hyperedge  $e = \{u_1, \dots, u_{2\ell}\}$  to  $G'$  for all distinct  $u_1, \dots, u_{2\ell} \in V$  that induce a  $2\ell$ -clique in  $G$ .
- Run  $\mathbb{A}$  on  $(G', \rho)$ . Let  $S$  be  $\mathbb{A}$ 's output.
- Use brute force to find a maximum balanced biclique in  $S$  and output it.

Notice that algorithm  $\mathbb{B}$  runs in time  $(f(2k) + 2^{O(k)}) \cdot n^{O(\ell)} = (f(2k) + 2^{O(k)}) \cdot n^{o(k)}$  as desired, where the term  $2^{O(k)}$  comes from the last step.

Next, we claim that, when  $k$  is sufficiently large, the output biclique has size at least  $t := \lfloor k/8 \rfloor$ , which would give us the desired  $O(1)$ -approximation ratio. Suppose for the sake of contradiction that this is not true, i.e. that the induced graph  $G[S]$  is  $K_{t,t}$ -free.

For any sufficiently large  $k$ , we have  $\ell < t$ . Hence, we may apply Lemma 16, which gives the following upper bound on the number of not-necessarily induced copies of  $K_{\ell,\ell}$  in  $G[S]$ :

$$\left(2 \cdot e^{-\frac{\ell^2}{16t}}\right) \cdot \binom{2k}{\ell} \binom{2k - \ell}{\ell} \leq e^{-\Omega(k/g(k)^{0.2})} \cdot \binom{2k}{\ell} \binom{2k - \ell}{\ell}.$$

However, since each hyperedge  $e = \{u_1, \dots, u_{2\ell}\}$  corresponds to  $\binom{2\ell}{\ell}$  copies of  $K_{\ell,\ell}$ , the number of hyperedges fully contained in  $S$  is thus at most

$$e^{-\Omega(k/g(k)^{0.2})} \cdot \binom{2k}{\ell} \binom{2k - \ell}{\ell} / \binom{2\ell}{\ell} = e^{-\Omega(k/g(k)^{0.2})} \cdot \binom{2k}{2\ell},$$

which is less than  $2^{-\rho/g(\rho)} \cdot \binom{2k}{2\ell}$  for any sufficiently large  $k$ . This contradicts the approximation guarantee of  $\mathbb{A}$  since the optimal solution (i.e. the  $2k$ -clique) contains  $\binom{2k}{2\ell}$  hyperedges.  $\blacktriangleleft$

---

#### References

- 1 Amir Abboud and Greg Bodwin. Reachability preservers: New extremal bounds and approximation algorithms. In *SODA*, pages 1865–1883, 2018. doi:10.1137/1.9781611975031.122.
- 2 Michael Alekhnovich. More on average case vs approximation complexity. *Computational Complexity*, 20(4):755–786, 2011.
- 3 Noga Alon, Sanjeev Arora, Rajsekar Manokaran, Dana Moshkovitz, and Omri Weinstein. Inapproximability of densest  $\kappa$ -subgraph from average case hardness, 2011.
- 4 Noga Alon, Uriel Feige, Avi Wigderson, and David Zuckerman. Derandomized graph products. *Comput. Complex.*, 5(1):60–75, 1995. doi:10.1007/BF01277956.
- 5 Noga Alon, Raphael Yuster, and Uri Zwick. Finding and counting given length cycles. *Algorithmica*, 17(3):209–223, 1997. doi:10.1007/BF02523189.

- 6 Benny Applebaum. Pseudorandom generators with long stretch and low locality from random local one-way functions. *SIAM J. Comput.*, 42(5):2008–2037, 2013. doi:10.1137/120884857.
- 7 Benny Applebaum, Boaz Barak, and Avi Wigderson. Public-key cryptosystem from different assumptions. In *STOC*, pages 171–180, 2010.
- 8 Sanjeev Arora, Boaz Barak, Markus Brunnnermeier, and Rong Ge. Computational complexity and information asymmetry in financial products (extended abstract). In *ICS*, pages 49–65, 2010. URL: <http://conference.iis.tsinghua.edu.cn/ICS2010/content/papers/5.html>.
- 9 Yakov Babichenko, Christos H. Papadimitriou, and Aviad Rubinfeld. Can almost everybody be almost happy? In *ITCS*, pages 1–9, 2016. doi:10.1145/2840728.2840731.
- 10 Maria-Florina Balcan, Christian Borgs, Mark Braverman, Jennifer T. Chayes, and Shang-Hua Teng. Finding endogenously formed communities. In *SODA*, pages 767–783, 2013. doi:10.1137/1.9781611973105.55.
- 11 Boaz Barak, Samuel B. Hopkins, Jonathan A. Kelner, Pravesh K. Kothari, Ankur Moitra, and Aaron Potechin. A nearly tight sum-of-squares lower bound for the planted clique problem. *SIAM J. Comput.*, 48(2):687–735, 2019. doi:10.1137/17M1138236.
- 12 Boaz Barak, Guy Kindler, and David Steurer. On the optimality of semidefinite relaxations for average-case and generalized constraint satisfaction. In *ITCS*, pages 197–214, 2013.
- 13 Siddharth Barman. Approximating Nash equilibria and dense subgraphs via an approximate version of Carathéodory’s theorem. *SIAM J. Comput.*, 47(3):960–981, 2018. doi:10.1137/15M1050574.
- 14 Mihir Bellare, Shafi Goldwasser, Carsten Lund, and A. Russell. Efficient probabilistically checkable proofs and applications to approximations. In *STOC*, pages 294–304, 1993. doi:10.1145/167088.167174.
- 15 Piotr Berman, Arnab Bhattacharyya, Konstantin Makarychev, Sofya Raskhodnikova, and Grigory Yaroslavtsev. Approximation algorithms for spanner problems and directed steiner forest. *Inf. Comput.*, 222:93–107, 2013. doi:10.1016/j.ic.2012.10.007.
- 16 Piotr Berman and Georg Schnitger. On the complexity of approximating the independent set problem. In *STACS*, pages 256–268, 1989. doi:10.1007/BFb0028990.
- 17 Quentin Berthet and Philippe Rigollet. Complexity theoretic lower bounds for sparse principal component detection. In *COLT*, pages 1046–1066, 2013. URL: <http://proceedings.mlr.press/v30/Berthet13.html>.
- 18 Tengyao Wang, Quentin Berthet, and Richard J. Samworth. Statistical and computational trade-offs in estimation of sparse principal components. *The Annals of Statistics*, 2016.
- 19 Amey Bhangale, Rajiv Gandhi, Mohammad Taghi Hajiaghayi, Rohit Khandekar, and Guy Kortsarz. Bicovering: Covering edges with two small subsets of vertices. In *ICALP*, pages 6:1–6:12, 2016. doi:10.4230/LIPIcs.ICALP.2016.6.
- 20 Umang Bhaskar, Yu Cheng, Young Kun Ko, and Chaitanya Swamy. Hardness results for signaling in bayesian zero-sum and network routing games. In *EC*, pages 479–496, 2016. doi:10.1145/2940716.2940753.
- 21 Aditya Bhaskara, Moses Charikar, Eden Chlamtac, Uriel Feige, and Aravindan Vijayaraghavan. Detecting high log-densities: an  $O(n^{1/4})$  approximation for densest  $k$ -subgraph. In *STOC*, pages 201–210, 2010. doi:10.1145/1806689.1806719.
- 22 Aditya Bhaskara, Moses Charikar, Aravindan Vijayaraghavan, Venkatesan Guruswami, and Yuan Zhou. Polynomial integrality gaps for strong SDP relaxations of densest  $k$ -subgraph. In *SODA*, pages 388–405, 2012. doi:10.1137/1.9781611973099.34.
- 23 Arnab Bhattacharyya, Suprovat Ghoshal, Karthik C. S., and Pasin Manurangsi. Parameterized intractability of even set and shortest vector problem from Gap-ETH. In *ICALP*, pages 17:1–17:15, 2018. doi:10.4230/LIPIcs.ICALP.2018.17.
- 24 Markus Bläser, Balagopal Komarath, and Karteeek Sreenivasaiah. Graph pattern polynomials. In *FSTTCS*, pages 18:1–18:13, 2018. doi:10.4230/LIPIcs.FSTTCS.2018.18.
- 25 Mark Braverman, Young Kun-Ko, Aviad Rubinfeld, and Omri Weinstein. ETH hardness for densest- $k$ -subgraph with perfect completeness. In *SODA*, pages 1326–1341, 2017.

- 26 Mark Braverman, Young Kun-Ko, and Omri Weinstein. Approximating the best Nash equilibrium in  $n^{o(\log n)}$ -time breaks the exponential time hypothesis. In *SODA*, pages 970–982, 2015. doi:10.1137/1.9781611973730.66.
- 27 Matthew Brennan and Guy Bresler. Optimal average-case reductions to sparse PCA: from weak assumptions to strong hardness. In *COLT*, pages 469–470, 2019. URL: <http://proceedings.mlr.press/v99/brennan19b.html>.
- 28 Matthew Brennan, Guy Bresler, Samuel B Hopkins, Jerry Li, and Tselil Schramm. Statistical query algorithms and low-degree tests are almost equivalent. *arXiv preprint*, 2020. arXiv: 2009.06107.
- 29 Matthew Brennan, Guy Bresler, and Wasim Huleihel. Reducibility and computational lower bounds for problems with planted sparse structure. In *COLT*, pages 48–166, 2018. URL: <http://proceedings.mlr.press/v75/brennan18a.html>.
- 30 Liming Cai and David W. Juedes. On the existence of subexponential parameterized algorithms. *J. Comput. Syst. Sci.*, 67(4):789–807, 2003. doi:10.1016/S0022-0000(03)00074-6.
- 31 Parinya Chalermsook, Marek Cygan, Guy Kortsarz, Bundit Laekhanukit, Pasin Manurangsi, Danupon Nanongkai, and Luca Trevisan. From Gap-ETH to FPT-Inapproximability: Clique, dominating set, and more. In *FOCS*, pages 743–754, 2017. doi:10.1109/FOCS.2017.74.
- 32 Moses Charikar. Greedy approximation algorithms for finding dense components in a graph. In *APPROX*, pages 84–95, 2000. doi:10.1007/3-540-44436-X\_10.
- 33 Moses Charikar, Chandra Chekuri, To-Yat Cheung, Zuo Dai, Ashish Goel, Sudipto Guha, and Ming Li. Approximation algorithms for directed steiner problems. *J. Algorithms*, 33(1):73–91, 1999. doi:10.1006/jagm.1999.1042.
- 34 Moses Charikar, MohammadTaghi Hajiaghayi, and Howard J. Karloff. Improved approximation algorithms for label cover problems. *Algorithmica*, 61(1):190–206, 2011. doi: 10.1007/s00453-010-9464-3.
- 35 Chandra Chekuri, Guy Even, Anupam Gupta, and Danny Segev. Set connectivity problems in undirected graphs and the directed steiner network problem. *ACM Trans. Algorithms*, 7(2):18:1–18:17, 2011. doi:10.1145/1921659.1921664.
- 36 Rajesh Chitnis, Andreas Emil Feldmann, and Pasin Manurangsi. Parameterized approximation algorithms for bidirected steiner network problems. In *ESA*, pages 20:1–20:16, 2018. doi: 10.4230/LIPIcs.ESA.2018.20.
- 37 Eden Chlamtác, Michael Dinitz, Christian Konrad, Guy Kortsarz, and George Rabanca. The densest  $k$ -subhypergraph problem. *SIAM J. Discret. Math.*, 32(2):1458–1477, 2018. doi:10.1137/16M1096402.
- 38 Eden Chlamtác, Michael Dinitz, Guy Kortsarz, and Bundit Laekhanukit. Approximating spanners and directed steiner forest: Upper and lower bounds. In *SODA*, pages 534–553, 2017. doi:10.1137/1.9781611974782.34.
- 39 Eden Chlamtac, Michael Dinitz, and Robert Krauthgamer. Everywhere-sparse spanners via dense subgraphs. In *FOCS*, pages 758–767, 2012. doi:10.1109/FOCS.2012.61.
- 40 Eden Chlamtác, Michael Dinitz, and Yury Makarychev. Minimizing the union: Tight approximations for small set bipartite vertex expansion. In *SODA*, pages 881–899, 2017. doi:10.1137/1.9781611974782.56.
- 41 Eden Chlamtác and Pasin Manurangsi. Sherali-adams integrality gaps matching the log-density threshold. In *APPROX*, pages 10:1–10:19, 2018. doi:10.4230/LIPIcs.APPROX-RANDOM.2018.10.
- 42 Eden Chlamtác, Pasin Manurangsi, Dana Moshkovitz, and Aravindan Vijayaraghavan. Approximation algorithms for label cover and the log-density threshold. In *SODA*, pages 900–919, 2017. doi:10.1137/1.9781611974782.57.
- 43 Radu Curticapean, Holger Dell, and Dániel Marx. Homomorphisms are a good basis for counting small subgraphs. In *STOC*, pages 210–223, 2017. doi:10.1145/3055399.3055502.



- 44 Radu Curticapean and Dániel Marx. Complexity of counting subgraphs: Only the boundedness of the vertex-cover number counts. In *FOCS*, pages 130–139, 2014. doi:10.1109/FOCS.2014.22.
- 45 Marek Cygan, Pawel Komosa, Daniel Lokshtanov, Michal Pilipczuk, Marcin Pilipczuk, and Saket Saurabh. Randomized contractions meet lean decompositions. *CoRR*, abs/1810.06864, 2018. arXiv:1810.06864.
- 46 Marek Cygan, Marcin Pilipczuk, and Michal Pilipczuk. Known algorithms for edge clique cover are probably optimal. *SIAM J. Comput.*, 45(1):67–83, 2016. doi:10.1137/130947076.
- 47 Søren Dahlgaard, Mathias Bæk Tejs Knudsen, and Morten Stöckel. Finding even cycles faster via capped  $k$ -walks. In *STOC*, pages 112–120, 2017. doi:10.1145/3055399.3055459.
- 48 Mina Dalirrooyfard, Thuy Duong Vuong, and Virginia Vassilevska Williams. Graph pattern detection: hardness for all induced patterns and faster non-induced cycles. In *STOC*, pages 1167–1178, 2019. doi:10.1145/3313276.3316329.
- 49 Amit Daniely. Complexity theoretic limitations on learning halfspaces. In *STOC*, pages 105–117, 2016. doi:10.1145/2897518.2897520.
- 50 Amit Daniely and Shai Shalev-Shwartz. Complexity theoretic limitations on learning DNF’s. In *COLT*, pages 815–830, 2016. URL: <http://proceedings.mlr.press/v49/daniely16.html>.
- 51 Michael Dinitz, Guy Kortsarz, and Zeev Nutov. Improved approximation algorithm for steiner  $k$ -forest with nearly uniform weights. *ACM Trans. Algorithms*, 13(3):40:1–40:16, 2017. doi:10.1145/3077581.
- 52 Irit Dinur and Pasin Manurangsi. ETH-hardness of approximating 2-CSPs and directed steiner network. In *ITCS*, pages 36:1–36:20, 2018. doi:10.4230/LIPIcs.ITCS.2018.36.
- 53 Yevgeniy Dodis and Sanjeev Khanna. Design networks with bounded pairwise distance. In *STOC*, pages 750–759, 1999. doi:10.1145/301250.301447.
- 54 Rodney G. Downey and Michael R. Fellows. Fixed-parameter tractability and completeness II: on completeness for  $W[1]$ . *Theor. Comput. Sci.*, 141(1&2):109–131, 1995. doi:10.1016/0304-3975(94)00097-3.
- 55 Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Springer, 2013.
- 56 Paul Erdős and Alfréd Rényi. Asymmetric graphs. *Acta Mathematica Academiae Scientiarum Hungarica*, 14(3-4):295–315, 1963.
- 57 Uriel Feige. Relations between average case complexity and approximation complexity. In *STOC*, pages 534–543, 2002. doi:10.1145/509907.509985.
- 58 Uriel Feige, Guy Kortsarz, and David Peleg. The dense  $k$ -subgraph problem. *Algorithmica*, 29(3):410–421, 2001. doi:10.1007/s004530010050.
- 59 Uriel Feige and Robert Krauthgamer. The probable value of the lovász–schrijver relaxations for maximum independent set. *SIAM J. Comput.*, 32(2):345–370, 2003. doi:10.1137/S009753970240118X.
- 60 Uriel Feige and Michael Seltser. On the densest  $k$ -subgraph problem. Technical report, Weizmann Institute of Science, Rehovot, Israel, 1997.
- 61 Moran Feldman, Guy Kortsarz, and Zeev Nutov. Improved approximation algorithms for directed steiner forest. *J. Comput. Syst. Sci.*, 78(1):279–292, 2012. doi:10.1016/j.jcss.2011.05.009.
- 62 Vitaly Feldman, Elena Grigorescu, Lev Reyzin, Santosh S. Vempala, and Ying Xiao. Statistical algorithms and a lower bound for detecting planted cliques. *J. ACM*, 64(2):8:1–8:37, 2017. doi:10.1145/3046674.
- 63 Chao Gao, Zongming Ma, and Harrison H Zhou. Sparse CCA: adaptive estimation and computational barriers. *The Annals of Statistics*, 2017.
- 64 Jiong Guo, Rolf Niedermeier, and Ondrej Suchý. Parameterized complexity of arc-weighted directed steiner problems. *SIAM J. Discret. Math.*, 25(2):583–599, 2011. doi:10.1137/100794560.


- 65 Anupam Gupta, Mohammad Taghi Hajiaghayi, Viswanath Nagarajan, and R. Ravi. Dial a ride from  $k$ -forest. *ACM Trans. Algorithms*, 6(2):41:1–41:21, 2010. doi:10.1145/1721837.1721857.
- 66 Bruce E. Hajek, Yihong Wu, and Jiaming Xu. Computational lower bounds for community detection on random graphs. In *COLT*, pages 899–928, 2015. URL: <http://proceedings.mlr.press/v40/Hajek15.html>.
- 67 Mohammad Taghi Hajiaghayi and Kamal Jain. The prize-collecting generalized steiner tree problem via a new approach of primal-dual schema. In *SODA*, pages 631–640, 2006. URL: <http://dl.acm.org/citation.cfm?id=1109557.1109626>.
- 68 Elad Hazan and Robert Krauthgamer. How hard is it to approximate the best Nash equilibrium? *SIAM J. Comput.*, 40(1):79–91, 2011. doi:10.1137/090766991.
- 69 Russell Impagliazzo and Ramamohan Paturi. On the complexity of  $k$ -SAT. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001. doi:10.1006/jcss.2000.1727.
- 70 Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001. doi:10.1006/jcss.2001.1774.
- 71 Mark Jerrum. Large cliques elude the metropolis process. *Random Struct. Algorithms*, 3(4):347–360, 1992. doi:10.1002/rsa.3240030402.
- 72 David S. Johnson. The NP-completeness column: An ongoing guide. *J. Algorithms*, 8(5):438–448, September 1987.
- 73 Richard Karp. Probabilistic analysis of some combinatorial search problems. *Algorithms and Complexity: New Directions and Recent Results*, 1976.
- 74 Richard M. Karp. Reducibility among combinatorial problems. In *Proceedings of a symposium on the Complexity of Computer Computations*, pages 85–103, 1972. doi:10.1007/978-1-4684-2001-2\_9.
- 75 Subhash Khot. Ruling out PTAS for graph min-bisection, dense  $k$ -subgraph, and bipartite clique. *SIAM J. Comput.*, 36(4):1025–1071, 2006. doi:10.1137/S0097539705447037.
- 76 Guy Kortsarz and David Peleg. On choosing a dense subgraph (extended abstract). In *34th Annual Symposium on Foundations of Computer Science, Palo Alto, California, USA, 3-5 November 1993*, pages 692–701, 1993.
- 77 Tamás Kővári, Vera T Sós, and Pál Turán. On a problem of Zarankiewicz. In *Colloquium Mathematicum*, volume 3, pages 50–57, 1954.
- 78 Mirosław Kowaluk, Andrzej Lingas, and Eva-Marta Lundell. Counting and detecting small subgraphs via equations. *SIAM J. Discret. Math.*, 27(2):892–909, 2013. doi:10.1137/110859798.
- 79 James R. Lee, Prasad Raghavendra, and David Steurer. Lower bounds on the size of semidefinite programming relaxations. In *STOC*, pages 567–576, 2015. doi:10.1145/2746539.2746599.
- 80 Bingkai Lin. The parameterized complexity of  $k$ -biclique. In *SODA*, pages 605–615, 2015.
- 81 Andrea Lincoln and Nikhil Vyas. Algorithms and lower bounds for cycles and walks: Small space and sparse graphs. In *ITCS*, pages 11:1–11:17, 2020. doi:10.4230/LIPIcs.ITCS.2020.11.
- 82 Daniel Lokshtanov, Dániel Marx, and Saket Saurabh. Lower bounds based on the exponential time hypothesis. *Bull. EATCS*, 105:41–72, 2011. URL: <http://eatcs.org/beatcs/index.php/beatcs/article/view/92>.
- 83 Daniel Lokshtanov, M. S. Ramanujan, Saket Saurabh, and Meirav Zehavi. Parameterized complexity and approximability of directed odd cycle transversal. *CoRR*, abs/1704.04249, 2017. arXiv:1704.04249.
- 84 Pasin Manurangsi. Almost-polynomial ratio ETH-hardness of approximating densest  $k$ -subgraph. In *STOC*, pages 954–961, 2017. doi:10.1145/3055399.3055412.
- 85 Pasin Manurangsi. Inapproximability of maximum edge biclique, maximum balanced biclique and minimum  $k$ -cut from the small set expansion hypothesis. In *ICALP*, pages 79:1–79:14, 2017. doi:10.4230/LIPIcs.ICALP.2017.79.
- 86 Pasin Manurangsi and Dana Moshkovitz. Approximating dense max 2-CSPs. In *APPROX*, pages 396–415, 2015. doi:10.4230/LIPIcs.APPROX-RANDOM.2015.396.



- 87 Pasin Manurangsi and Aviad Rubinfeld. Inapproximability of VC dimension and Littlestone's dimension. In *COLT*, pages 1432–1460, 2017. URL: <http://proceedings.mlr.press/v65/manurangsi17a.html>.
- 88 R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. Network motifs: Simple building blocks of complex networks. *Science*, 298(5594):824–827, 2002.
- 89 Jaroslav Nesetril and Svatopluk Poljak. On the complexity of the subgraph problem. *Commentationes Mathematicae Universitatis Carolinae*, 26:415–419, 1985.
- 90 Prasad Raghavendra and David Steurer. Graph expansion and the unique games conjecture. In *STOC*, pages 755–764, 2010. doi:10.1145/1806689.1806792.
- 91 Aviad Rubinfeld. Settling the complexity of computing approximate two-player Nash equilibria. In *FOCS*, pages 258–265, 2016. doi:10.1109/FOCS.2016.35.
- 92 Aviad Rubinfeld. Detecting communities is hard (and counting them is even harder). In *ITCS*, pages 42:1–42:13, 2017. doi:10.4230/LIPIcs.ITCS.2017.42.
- 93 Danny Segev and Gil Segev. Approximate  $k$ -steiner forests via the lagrangian relaxation technique with internal preprocessing. *Algorithmica*, 56(4):529–549, 2010. doi:10.1007/s00453-008-9189-8.
- 94 Johan Ugander, Lars Backstrom, and Jon M. Kleinberg. Subgraph frequencies: mapping the empirical and extremal geography of large graph collections. In *WWW*, pages 1307–1318, 2013. doi:10.1145/2488388.2488502.
- 95 Virginia Vassilevska Williams. On some fine-grained questions in algorithms and complexity, 2018. ICM survey.
- 96 Virginia Vassilevska Williams and Ryan Williams. Finding, minimizing, and counting weighted subgraphs. *SIAM J. Comput.*, 42(3):831–854, 2013. doi:10.1137/09076619X.
- 97 Raphael Yuster and Uri Zwick. Detecting short directed cycles using rectangular matrix multiplication and dynamic programming. In *SODA*, pages 254–260, 2004. URL: <http://dl.acm.org/citation.cfm?id=982792.982828>.
- 98 David Zuckerman. On unapproximable versions of NP-complete problems. *SIAM Journal on Computing*, 25(6):1293–1304, 1996.



# Sample Efficient Identity Testing and Independence Testing of Quantum States

Nengkun Yu 

Centre for Quantum Software and Information, Faculty of Engineering and Information Technology,  
University of Technology, Sydney, Australia  
nengkunyu@gmail.com

---

## Abstract

In this paper, we study the quantum identity testing problem, i.e., testing whether two given quantum states are identical, and quantum independence testing problem, i.e., testing whether a given multipartite quantum state is in tensor product form. For the quantum identity testing problem of  $\mathcal{D}(\mathbb{C}^d)$  system, we provide a deterministic measurement scheme that uses  $\mathcal{O}(\frac{d^2}{\epsilon^2})$  copies via independent measurements with  $d$  being the dimension of the state and  $\epsilon$  being the additive error. For the independence testing problem  $\mathcal{D}(\mathbb{C}^{d_1} \otimes \mathbb{C}^{d_2} \otimes \dots \otimes \mathbb{C}^{d_m})$  system, we show that the sample complexity is  $\tilde{\Theta}(\frac{\sum_{i=1}^m d_i}{\epsilon^2})$  via collective measurements, and  $\mathcal{O}(\frac{\sum_{i=1}^m d_i^2}{\epsilon^2})$  via independent measurements. If randomized choice of independent measurements are allowed, the sample complexity is  $\Theta(\frac{d^{3/2}}{\epsilon^2})$  for the quantum identity testing problem, and  $\tilde{\Theta}(\frac{\sum_{i=1}^m d_i^{3/2}}{\epsilon^2})$  for the quantum independence testing problem.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Quantum computation theory; Mathematics of computing  $\rightarrow$  Probability and statistics

**Keywords and phrases** Quantum property testing

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.11

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/1904.03218>.

**Funding** Nengkun Yu: N. Y. is supported by ARC Discovery Early Career Researcher Award DE180100156 and ARC Discovery Program DP210102449.

**Acknowledgements** We thank Youming Qiao for his helpful comments on the previous version of this manuscript. We thank Tongyang Li for pointing out relevant reference [38]. We thank Ryan O'Donnell and John Wright for telling us the Sanov's theorem and its relation to [44]. We are grateful for the reviewer to point out that using random independent measurement can provide a tight bound for quantum state certification.

## 1 Introduction

### 1.1 Classical Property Testing

The ability to test whether an unknown object satisfies a hypothetical model based on observed data plays a particularly important role in science [50]. Initially proposed by Rubinfeld and Sudan [60, 61] to test algebraic properties of polynomials, the concept of property testing has been extended to many objects of computer science: graphs, Boolean functions, and so on [41, 40]. Property testing and distribution testing are intricately connected. At the beginning of this century, Batu *et al.* introduced the problem of testing properties associated with discrete probability distributions [14, 15]. In other words, how many samples from a collection of probability distributions are needed to determine whether those distributions satisfy a particular property with high confidence? Over the past two decades, this area has become an extremely well-studied and successful branch of property testing due in part to



© Nengkun Yu;

licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 11; pp. 11:1–11:20



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

the ongoing data science revolution. Never have computationally-efficient algorithms, a.k.a., testers, that can identify and/or classify properties using as few samples as possible been in higher demand.

A direct approach to distribution property testing is to reconstruct the given distributions from sufficiently many samples. It is well known that, after taking  $\Theta(\epsilon^{-2} \cdot d)$  samples from a  $d$ -dimensional probability distribution  $p$ , the empirical distribution is, with high probability,  $\epsilon$ -close to  $p$  in total variance distance [29, pages 10 and 31]. Therefore, finding algorithms that use  $o(d)$  samples for testing problems is highly desirable. Surprisingly, algorithms using less number of samples than  $\Theta(\epsilon^{-2} \cdot d)$  exist for many important properties.

The equality, or identity, of distributions is a central problem in this branch of study, and one that is frequently revisited with different approaches due to its importance. In an important work [40], Goldreich and Ron found that the  $\ell_2$  norm can be estimated from  $O(\epsilon^{-2} \cdot \sqrt{d})$  samples. This led to an algorithm for uniformity testing, *i.e.*, to determine whether a probability distribution is a uniform using  $O(\epsilon^{-4} \cdot \sqrt{d})$  samples. Paninski [58] and Valiant and Valiant [63] showed that the complexity of uniformity is  $\Theta(\epsilon^{-2} \cdot \sqrt{d})$ . If one distribution is an arbitrary known distribution, Batu *et al.* [14, 13] presented an  $\ell_2$ -identity tester and used it to build an  $\ell_1$  estimator using  $O(\epsilon^{-2} \cdot \sqrt{d} \log d)$  samples; later in [65], Valiant and Valiant showed the sample complexity of this problem is  $\Theta(\epsilon^{-2} \cdot \sqrt{d})$ . If both distributions are unknown, Batu *et al.* provided a tester in [14] using  $O(\epsilon^{-8/3} \cdot d^{2/3} \log d)$  samples; In 2014, Chan *et al.*, in [23], showed the complexity of the identity testing is  $\Theta(\max(\epsilon^{-2} \cdot \sqrt{d}, \epsilon^{-4/3} \cdot d^{2/3}))$ .

The idea of identity testing has been extensively explored in studying other property testing problems. Independence testing and conditional independence testing are among the most important ones. In [13], Batu *et al* presented an independence tester for bipartite independence testing over  $[d_1] \times [d_2]$  with a sample complexity of  $\tilde{O}(d_1^{2/3} d_2^{1/3}) \cdot \text{Poly}(\epsilon^{-1})$ , for  $d_1 \geq d_2$ . Levi, Ron and Rubinfeld in [51] showed a lower bound  $\Omega(\sqrt{d_1 d_2})$  for all  $d_1 \geq d_2$  and  $\Omega(d_1^{2/3} d_2^{1/3})$  for  $d_1 = \Omega(d_2 \log d_2)$ . Acharya *et al.* [6] introduced a tester for multipartite independence testing over  $\times_{j=1}^m [d_j]$  with sample complexity  $O(\epsilon^{-2} \cdot \sqrt{\prod_{j=1}^m d_j} + \epsilon^{-2} \cdot \sum_{j=1}^m d_j)$ . In their important work [31], Diakonikolas and Kane demonstrated a unified approach to resolve the sample complexity of a wide variety of testing problems based on their alternative proof for identity testing. In particular, they showed that the sample complexity of independence testing is  $\Theta(\max_k \{\epsilon^{-2} \sqrt{\prod_{j=1}^m d_j}, \epsilon^{-4/3} \cdot d_k^{2/3} \cdot \prod_{j=1}^m d_j^{1/3}\})$ . Canonne *et al.* [22] initiated the study of the conditional independence within property testing framework. Notably, for the very important  $[2] \times [2] \times [n]$ , they showed that the sample complexity for this problem is  $\Theta(\max\{\epsilon^{-2} \cdot \sqrt{n}, \min\{\epsilon^{-1} \cdot n^{7/8}, \epsilon^{-8/7} \cdot n^{6/7}\}\})$ .

Besides the mentioned works, a very incomplete list of works of distributional property testing includes [12, 15, 68, 5, 64, 51, 46, 26, 32, 48, 66, 70, 30, 67, 28, 39, 33, 7, 21, 27, 34, 24], and two excellent surveys include more [59, 20].

## 1.2 Quantum Property Testing

Quantum property testing has been extensively studied. At this stage of development of quantum computation, testing the properties of new devices as they are built is a basic problem as illustrated in Montanaro and de Wolf's comprehensive survey [53]. A standard quantum device outputs some known  $d$ -dimensional (mixed) state  $\sigma \in \mathcal{D}(\mathbb{C}^d)$  but inevitably, the results are noisy such that the actual output state  $\rho \in \mathcal{D}(\mathbb{C}^d)$  is not equal  $\sigma$ , maybe not even close to. Similar to property testing with classical distributions, properties of  $\rho$  need to be verified by accessing the device, say,  $m$  times, to derive  $\rho^{\otimes m}$ .

Starting from the very basic problem of quantum state tomography, a fundamental problem is to decide how many copies of an unknown mixed quantum state  $\rho \in \mathcal{D}(\mathbb{C}^d)$  is necessary and sufficient to output a good approximation of  $\rho$  in trace distance, with high probability. This problem has been studied extensively since the birth of quantum information theory. The main-stream approach is through independent measurement, i.e., measurement on each copy of the state. A sequence of work [42, 36, 69, 49] is dedicated to showing that  $O(\epsilon^{-2} \cdot d^3)$  copies are sufficient in an  $\ell_1$  distance of no more than  $\epsilon$ . Haah *et al.* [44] showed that this is tight for independent measurement. For joint measurement, Haah *et al.* [44] proved that  $O(\epsilon^{-1} \cdot d^2 \log(\epsilon^{-1} \cdot d))$  copies are sufficient to obtain an infidelity of no more than  $\epsilon$ , which can be regarded as a quantum generalization of Sanov's theorem [62]. By combining the lower bound of [44] and upper bound of [56, 57], the sample complexity of state tomography with joint measurement is  $\tilde{\Theta}(\epsilon^{-2} \cdot d^2)$  in a  $\ell_1$  distance error of less than  $\epsilon$  with high probability.

A more direct approach to quantum property testing is to estimate  $\rho$  by sampling from  $\rho^{\otimes m}$ , which also means one could check any property of interest. However, like classical property testing, this idea is not optimal for a general property. One problem that has received much attention is quantum identity testing. Suppose we are given query access to two states  $\rho, \sigma \in \mathcal{D}(\mathbb{C}^d)$ , and we want to test whether they are equal or have a large  $\ell_1$  distance. For practical purposes, the results from cases where  $\sigma$  is a known pure state have been extensively studied, in the independent measurement setting [37, 25, 10]. [55] solved the problem, in the joint measurement setting, where  $\sigma$  is a maximally mixed state case by showing that  $\Theta(\epsilon^{-2} \cdot d)$  copies are necessary and sufficient. Importantly, the sample complexity of the general problem was proven to be  $\Theta(\epsilon^{-2} \cdot d)$  in [18] by providing an efficient  $\ell_2$  distance estimator between two unknown quantum states.

In [4], Aaronson initialized the study of the learnability of quantum state, whose goal is to output good estimations of a set of measurements simultaneously. In [1], Aaronson provided an efficient procedure of the quantum shadow tomography. A connection between quantum learning and differential privacy was established in [3]. In [2], the online learning of quantum states was studied.

Entanglement is a ubiquitous phenomenon in quantum information theory. A multipartite pure state  $|\psi\rangle \in (\mathbb{C}^d)^{\otimes m}$  is not entangled if it can be written as  $|\psi\rangle = \bigotimes_{j=1}^m |\psi_j\rangle$  for some  $|\psi_j\rangle \in \mathbb{C}^d$ . Pure entanglement testing was first discussed by Mintert *et al.* [52]. Harrow and Montanaro [45] subsequently proved that  $O(\epsilon^{-2})$  copies are sufficient and used that to study the quantum complexity theory. In [17], it was proved that  $\Omega(d^2/\epsilon^2)$  copies are necessary to test separability of quantum states in  $\mathbb{C}^d \otimes \mathbb{C}^d$  for not small  $\epsilon$ .

Acharya *et al.* [8] estimated the von Neumann entropy of general quantum states. Gross *et al.* [43] showed that “stabilizerness” can be tested efficiently. One research direction is to study the potential speed-up of distributional property testing using quantum algorithms where the distribution is given in the form of a quantum oracle [16, 38].

### 1.3 Measurement Schemes

A significant difference between quantum property testing and classical property testing is the way the objects are sampled. In classical property testing, each sample is output with a classical index according to the probability distribution and given a fixed number of samples, the output string obeys the product probability distribution. However, with quantum property testing, the sampling methods have much richer structures. This difference together with others prevents the potential to design algorithms for quantum property testing from ingenious ideas and techniques of distribution testing.

Measurement	Complexity	Dimension	Implementation
Joint	Low	$d^m$	Hard even in the future
Independent	Medium	$d$	Available in the future

Among the many available sampling methods for quantum property testing (given a fixed number of copies, says  $k$ , of the states  $\rho \in \mathcal{D}(\mathbb{C}^d)$ ), the two listed in Table 1 are of particular interests, i.e., joint measurement, and independent measurement. Joint measurement, the most general, allows arbitrary measurements of  $\mathbb{C}^{d^m}$ . Independent measurement only allows non-adaptive measurements on each copy of  $\rho$ , which results in,  $n$  measurements of  $\mathbb{C}^d$ .

Joint measurement has the potential to provide the optimal number of samples, but there are two caveats. “Optimal” joint measurement algorithms usually require an *exponential* number of copies of the quantum state to produce optimal results. They are also based on the assumption of noiseless, universal quantum computation on the *exponential* number of copies of the quantum state. For instance, the optimal tomography algorithms of  $k$ -qubit quantum state in [44, 56, 57] require a joint measurement on  $\Theta(\epsilon^{-2} \cdot k 2^{2k})$  qubits. Even in the future when quantum computers become a reality, implementing optimal joint measurement would be extremely hard given these conditions. General independent measurements are not feasible with currently-available technology. To implement a two-outcome measurement on the  $k$ -qubit system, one needs to implement a  $k + 1$  qubit unitary. Implementing a general  $k + 1$  qubit unitary requires a circuit consisting of at least  $\Omega(4^k)$  elementary gates, which could also be hard.

In this paper, we study the quantum identity testing problem, i.e., testing whether two given quantum states are identical, and the quantum independence testing problem, i.e., testing whether a given multipartite quantum state is in tensor product form. For the quantum identity testing problem of  $\mathcal{D}(\mathbb{C}^d)$  system, we provide a measurement scheme that uses  $\mathcal{O}(\frac{d^2}{\epsilon^2})$  copies via independent measurements with  $d$  being the dimension of the state and  $\epsilon$  being the additive error. For the independence testing problem  $\mathcal{D}(\mathbb{C}_1^d \otimes \mathbb{C}^{d_2} \otimes \dots \otimes \mathbb{C}^{d_m})$  system, we show that the sample complexity is  $\tilde{\Theta}(\frac{\sum_{i=1}^m d_i}{\epsilon^2})$  via collective measurements, and  $\mathcal{O}(\frac{\sum_{i=1}^m d_i^2}{\epsilon^2})$  via independent measurements. Further, we initialize the study of the property testing problems of classical-quantum states, motivated by the potential applications of classical-quantum states. Our main tool is a measurement that “preserves” the  $\ell_2$  distance, which invokes an immediate connection between quantum and classical property testing.

## 1.4 Our contributions

Identify whether two quantum states are equal or not is called *quantum identity testing problem*.

► **Problem 1.** *Given two unknown quantum mixed states  $\rho, \sigma \in \mathcal{D}(\mathbb{C}^d)$ , they satisfy either  $\rho = \sigma$  or  $\|\rho - \sigma\|_1 > \epsilon$  for a given  $\epsilon > 0$ . How many copies of  $\rho$  and  $\sigma$  are needed to distinguish these two cases, with high probability?*

This problem under joint measurement setting is solved in [18]. In this paper, we study this problem using independent measurement. To reach this goal, we observe the following lemma. It maintains interesting relations between the  $\ell_2$  distance of quantum states and the  $\ell_2$  distance of the generated corresponding probability distributions. Given that  $\ell_2$  distance plays a central role in classical property testing [31], our approach invokes an immediate connection between quantum and classical property testing. Previous research into quantum property testing has always been in isolation of classical property testing, whereas this scheme

opens up the potential to design quantum property tester from ingenious ideas and techniques of distribution testing. Further, this is a fixed measurement scheme that does not depend on the property to be tested, which makes our algorithms a perfect fit for implementation with current experiments.

► **Lemma 2.** *For  $d$  being power of 2, there is a measurement*

$$\mathcal{M} = (M_1, M_2, \dots, M_{d(d+1)}) : \mathcal{D}(\mathbb{C}^d) \mapsto \Delta(d(d+1))$$

*whose outcome lies in  $\Delta(d(d+1))$ , the  $d(d+1)$ -dimensional probability simplex, such that, for any quantum states  $\rho, \sigma \in \mathcal{D}(\mathbb{C}^d)$*

$$\|p - q\|_2 = \frac{\|\rho - \sigma\|_2}{d+1}, \quad \|p\|_2, \|q\|_2 \leq \frac{\sqrt{2}}{d+1}, \quad (1)$$

*where  $p = (p_1, p_2, \dots, p_{d(d+1)})$  and  $q = (q_1, q_2, \dots, q_{d(d+1)})$  with  $p_i = \text{Tr}(\rho M_i)$  and  $q_i = \text{Tr}(\sigma M_i)$ .*

We employ mutually unbiased bases (MUB) to construct such measurement. MUB in Hilbert space  $\mathbb{C}^d$  are two orthonormal bases  $\{|e_1\rangle, \dots, |e_d\rangle\}$  and  $\{|f_1\rangle, \dots, |f_d\rangle\}$  such that the square of the magnitude of the inner product between any basis states  $|e_j\rangle$  and  $|f_k\rangle$  equals the inverse of the dimension  $d$ . These bases are unbiased in the following sense: if a system is prepared in a state belonging to one of the bases, then all outcomes of the measurement with respect to the other basis are predicted to occur with equal probability.

For  $d = 2^n$ , there are  $2^n + 1$  mutually unbiased bases in  $\mathbb{C}^d$ . Therefore, the density matrices of these MUBs form a linear basis of  $\mathcal{D}(\mathbb{C}^d)$  in this case. Each measurement operator  $M_i$  is proportional to a density matrix of a MUB element. Therefore, after the measurement, there is no more information left because applying measurement in other MUB basis would output uniform distribution.

The upper bound of  $\ell_2$  norms of the output probability distribution is essential in designing an efficient quantum tester by lifting classical property tester because a small  $\ell_2$  norms ensures that the tester could use a smaller number of samples for the distributional identity testing problem as illustrated in [31], and distributional independence testing problem studied in [22].

Using Lemma 2 and the result of classical property testing, a tester using independent measurement for Problem 2 can be obtained as follows.

► **Theorem 3.** *For  $\rho, \sigma \in \mathcal{D}(\mathbb{C}^d)$ ,  $O(\epsilon^{-2} \cdot d^2)$  copies are sufficient to distinguish via deterministic independent measurements, with at least a  $\frac{2}{3}$  probability of success, the cases where  $\rho = \sigma$  from the cases where  $\|\rho - \sigma\|_1 > \epsilon$ .*

This is better than directly using the SWAP test which uses  $O(\frac{d^2}{\epsilon^4})$  copies, although the SWAP test is already a joint measurement.

Entanglement is a central feature in quantum information science. Certification of entanglement has received great amount of effort. This motivates us to study the following quantum independence testing problem.

► **Problem 4.** *Given an unknown quantum mixed states  $\rho \in \mathcal{D}(\mathbb{C}^{d_1} \otimes \mathbb{C}^{d_2} \otimes \dots \otimes \mathbb{C}^{d_n})$ , they satisfy either  $\rho = \sigma_1 \otimes \sigma_2 \otimes \dots \otimes \sigma_n$  for some  $\sigma_i \in \mathcal{D}(\mathbb{C}^{d_i})$  or for all  $\sigma_i \in \mathcal{D}(\mathbb{C}^{d_i})$ ,  $\|\rho - \sigma_1 \otimes \sigma_2 \otimes \dots \otimes \sigma_n\|_1 > \epsilon$  for a given  $\epsilon > 0$ . How many copies of  $\rho$  are needed to distinguish these two cases, with high probability?*



The above  $\ell_1$  identity testers for independent measurement together with the  $\ell_1$  identity tester of [18] for joint measurement enable us to derive the following result.

► **Theorem 5.** *The sample complexity of quantum independence testing problem for  $n$ -qubit quantum state is  $\Theta(\epsilon^{-2} \cdot 2^n)$ .*

*For general  $n$ -partite system  $\mathcal{D}(\mathbb{C}^{d_1} \otimes \mathbb{C}^{d_2} \otimes \dots \otimes \mathbb{C}^{d_n})$  with  $d_1 \geq d_2 \geq \dots \geq d_n$ , the sample complexity of quantum independence testing problem is  $\tilde{\Theta}(\epsilon^{-2} \cdot \prod_{i=1}^n d_i)$  with joint measurements; and  $O(\epsilon^{-2} \cdot \prod_{i=1}^n d_i^2)$  with deterministic independent measurements, where  $\tilde{\Theta}$  hides a factor between  $(\log^3 d_1 \cdot \log \log d_1)^{-1}$  and 1.*

In  $n$ -qubit system, the lower bound of the quantum independence for joint measurement comes from a reduction from determining whether a given state is a maximally mixed state.

In a general system, the lower bound is derived using an additional technique called dimension splitting which regards a  $d_1$ -dimensional system as  $\log d_1$  qubits system.

## 1.5 Other Results

It is widely believed that the fully-fledged quantum computer will be controlled through a classical system. Therefore, the data generated by quantum computers would be modeled by classical-quantum states, e.g., classical collections of quantum states. The importance of classical-quantum states also comes from its central role in studying quantum communication complexity [47, 9]. In classical property testing, Levi, Ron, and Rubinfeld initialized the study of property testing of collections of distributions in their pioneering work [51]. This motivates us to study the property testing problems of classical-quantum states.

In the query model, there are  $m$  states  $\rho_1, \rho_2, \dots, \rho_m$ . We can choose  $1 \leq i \leq n$  to obtain a copy of  $\rho_i$ . A motivation of studying this model is the quantum state preparation. Suppose there are different ways of generating a quantum state. We want to know whether these methods all work well. This problem can be formulated as *the independence testing of collections of quantum states*.

► **Problem 6.** *Given unknown quantum mixed states  $\rho_1, \rho_2, \dots, \rho_m \in \mathcal{D}(\mathbb{C}^{d_1} \otimes \mathbb{C}^{d_2} \otimes \dots \otimes \mathbb{C}^{d_n})$  and a given distribution  $p = (p_1, p_2, \dots, p_m)$ , they satisfy either there exist  $\sigma_{k,i} \in \mathcal{D}(\mathbb{C}^{d_k})$  for  $1 \leq k \leq n$  such that for all  $1 \leq i \leq m$   $\rho_i = \otimes_{k=1}^n \sigma_{k,i}$ , or for any  $\sigma_{k,i} \in \mathcal{D}(\mathbb{C}^{d_k})$ ,  $\sum_{i=1}^m p_i \|\rho_i - \otimes_{k=1}^n \sigma_{k,i}\|_1 > \epsilon$ , for a given  $\epsilon > 0$ . How many queries are needed to distinguish these two cases, with high probability?*

Combing the framework in [31] and our independence testers, we obtain

► **Theorem 7.** *The sample complexity of the independence testing of collections of quantum states is  $\tilde{\Theta}(\epsilon^{-2} \cdot d)$  with joint measurement;  $O(\epsilon^{-2} \cdot d^2)$  with deterministic independent measurement.*

Like their classical counterparts, the complexity does not depend on the number of states  $n$ . Similarly, this idea can be used for the independence testing of collections of quantum states.

In further work, we explore the problem of testing conditional independence with classical-quantum-quantum states. This question naturally arises in studying distributed quantum computing. One typical example is environment assisted entanglement distribution. Suppose  $\rho_{ABC}$  is a tripartite state. We want to reach the goal of sharing a bipartite state  $\sigma_{AB}$ .  $C$  should perform a measurement on its system, now the state becomes classical-quantum-quantum.

► **Problem 8.** *Given an unknown classical-quantum-quantum mixed states*

$$\rho_{ABC} = \sum_{i=1}^m p_m \rho_{AB,i} \otimes |i\rangle\langle i| \in \mathcal{D}(\mathbb{C}^{d_1} \otimes \mathbb{C}^{d_2}) \otimes \Delta(C)$$

with  $\Delta(C)$  being the probabilistic simplex of  $C$  and  $|C| = m$  and  $|i\rangle$  being a basis of  $C$ , we want to distinguish whether  $\rho_{ABC}$  is conditional independence, that is  $\rho = \sum_{i=1}^m p_m \rho_{A,i} \otimes \rho_{B,i} \otimes |i\rangle\langle i|$ , or for any conditional independent classical-quantum-quantum state  $\sigma_{ABC}$   $\|\rho - \sigma\|_1 > \epsilon$ , for a given  $\epsilon > 0$ . How many queries are needed to distinguish these two cases, with high probability?

This problem is a generalization of the independence testing of collections of quantum states in the sense that the prior coefficient of the  $\ell_1$  distance is not given explicitly but may be approximated through sampling. One motivation for studying this problem is a simplified version of the conditional independence of general tripartite quantum states, which a fundamental concept in theoretical physics and quantum information theory.

More specifically, we modify the  $\ell_2$  estimator developed in [18] for joint measurement and develop a finer  $\ell_2$  estimator for independent measurement. Then we plug that estimator into the classical conditional independence testing framework developed in [22].

► **Theorem 9.** *For classical-quantum-quantum state  $\rho_{ABC} \in \mathcal{D}(\mathbb{C}^{d_1} \otimes \mathbb{C}^{d_2}) \otimes \Delta(C)$ , the sample complexity of testing whether  $A$  and  $B$  are conditionally independent given  $C$  is*

- $O(\max\{\frac{\sqrt{nd_1d_2}}{\epsilon^2}, \min\{\frac{d_1^{\frac{4}{7}}d_2^{\frac{4}{8}}n^{\frac{6}{7}}}{\epsilon^{\frac{6}{7}}}, \frac{\sqrt{d_1d_2}n^{\frac{7}{8}}}{\epsilon}\}\})$  with joint measurement; and
- $O(\max\{\frac{\sqrt{nd_1^2d_2^2}}{\epsilon^2}, \min\{\frac{d_1^{\frac{6}{7}}d_2^{\frac{6}{8}}n^{\frac{6}{7}}}{\epsilon^{\frac{6}{7}}}, \frac{d_1^{\frac{3}{4}}d_2^{\frac{3}{4}}n^{\frac{7}{8}}}{\epsilon}\}\})$  with independent measurement.

## 1.6 Organization of this paper

Section 2 recalls the basic definitions of distance with discrete distributions and quantum states and presents some formal tools from earlier work that are used here. In Section 3, we state technical lemmata about the independence and conditional independence of quantum states. Section 4 demonstrates Lemma 2. Section 5 contains the results of identity testing and Theorems 3. In Section 6, we discuss the advantage of using random choice of independent measurements. Detail proofs of Lemmata, Theorem 7 and Theorem 9 can be found in the full version [71].

## 2 Preliminaries

This section begins with some standard notations and definitions used throughout the paper.

### 2.1 Basic facts for probability distributions

For  $m \in \mathbb{N}$ ,  $[m]$  denotes the set  $\{1, \dots, m\}$ , and  $\log$  denotes the binary logarithm. A probability distribution over discrete domain  $\Omega$  is a function  $p : \Omega \mapsto [0, 1]$  such that  $\sum_{\omega \in \Omega} p(\omega) = 1$ .  $|\Omega|$  is the cardinality of set  $\Omega$ .  $\Delta(\Omega)$  denotes the set of probability distributions over  $\Omega$ , i.e., the probability simplex of  $\Omega$ . The marginal distributions  $p_1 \in \Delta(A)$  and  $p_2 \in \Delta(B)$  of a bipartite distribution  $p_{1,2} \in \Delta(A \times B)$  can be defined as  $p_1(a) = \sum_{b \in B} p_{1,2}(a, b)$ ,  $p_1(b) = \sum_{a \in A} p_{1,2}(a, b)$ . The product distribution  $q_1 \otimes q_2$  of distributions  $q_1 \in \Delta(A)$  and  $q_2 \in \Delta(B)$  can be defined as  $[q_1 \otimes q_2](a, b) = q_1(a)q_2(b)$ , for every  $(a, b) \in A \times B$ .

The  $\ell_1$  distance between two distributions  $p, q \in \Delta(\Omega)$  is  $\|p - q\|_1 = \sum_{\omega \in \Omega} |p(\omega) - q(\omega)|$ , and their  $\ell_2$  distance is  $\|p - q\|_2 = \sqrt{\sum_{\omega \in \Omega} (p(\omega) - q(\omega))^2}$ .

## 2.2 Basic quantum mechanics

An isolated physical system is associated with a Hilbert space, which is called the *state space*. A *pure state* of a quantum system is a normalized vector in its state space, and a *mixed state* is represented by a density operator on the state space. Here, a density operator  $\rho$  on  $d$ -dimensional Hilbert space  $\mathbb{C}^d$  is a semi-definite positive linear operator such that  $\text{Tr}(\rho) = 1$ . We let

$$\mathcal{D}(\mathbb{C}^d) = \{\rho : \rho \text{ is } d\text{-dimensional density operator of } \mathbb{C}^d\}$$

denote the set of quantum states.

The reduced quantum state of a bipartite mixed state  $\rho_{1,2} \in \mathcal{D}(\mathbb{C}^{d_1} \otimes \mathbb{C}^{d_2})$  on the second system is the density operators  $\rho_2 := \text{Tr}_1 \rho_{1,2} = \sum_i \langle i|A|i\rangle$ , where  $\{|i\rangle\}$  is the orthonormal basis of  $\mathbb{C}^{d_1}$ . The partial trace of  $\rho_1 := \text{Tr}_2 \rho_{1,2}$  can be similarly defined, note that partial trace functions are also independent of the selected orthonormal basis. This definition can be directly generalized into multipartite quantum states.

## 2.3 Quantum measurement

A positive-operator valued measure (POVM) is a measure whose values are non-negative self-adjoint operators in a Hilbert space  $\mathbb{C}^d$ , which is described by a collection of matrices  $\{M_i\}$  with  $M_i \geq 0$  and  $\sum_i M_i = I_d$ . If the state of a quantum system was  $\rho$  immediately before measurement  $\{M_i\}$  was performed on it, then the probability of that result  $i$  recurring is  $p(i) = \text{Tr}(M_i \rho)$ .

## 2.4 $\ell_1$ distance

$\ell_1$  distance is used to characterize the difference between quantum states. The  $\ell_1$  distance between  $\rho$  and  $\sigma$  is defined as  $\|\rho - \sigma\|_1 \equiv \text{Tr}|\rho - \sigma|$  where  $|A| \equiv \sqrt{A^\dagger A}$  is the positive square root of  $A^\dagger A$ .

Given a general operator  $A$ , the  $\ell_1$  norm is defined as  $\|A\|_1 = \text{Tr}|A|$ . And Lemma 10 always applies:

► **Lemma 10** ([54]). *The  $\ell_1$  distance is decreasing under partial trace. That is*

$$\|\rho_1 - \sigma_1\|_1, \|\rho_2 - \sigma_2\|_1 \leq \|\rho_{1,2} - \sigma_{1,2}\|_1.$$

Their  $\ell_2$  distance is defined as  $\|\rho - \sigma\|_2 = \sqrt{\text{Tr}(\rho - \sigma)^2}$ . For  $\rho, \sigma \in \mathcal{D}(\mathbb{C}^d)$ , we have the following relation between  $\ell_1$  and  $\ell_2$  distances,  $\|\rho - \sigma\|_2 \leq \|\rho - \sigma\|_1 \leq \sqrt{d} \|\rho - \sigma\|_2$ . Given a subset  $\mathcal{P} \subsetneq \mathcal{D}(\mathbb{C}^d)$ , the  $\ell_1$  distance between  $\rho$  and  $\mathcal{P}$  is defined as  $\|\rho - \mathcal{P}\|_1 = \inf_{\sigma \in \mathcal{P}} \|\rho - \sigma\|_1$ . If  $\|\rho - \mathcal{P}\|_1 > \epsilon$ , we say that  $\rho$  is  $\epsilon$ -far from  $\mathcal{P}$ ; otherwise, it is  $\epsilon$ -close.

## 2.5 Mutually unbiased bases

In quantum information theory, mutually unbiased bases (MUB) in  $d$ -dimensional Hilbert space are two orthonormal bases  $\{|e_1\rangle, \dots, |e_d\rangle\}$  and  $\{|f_1\rangle, \dots, |f_d\rangle\}$  such that the square of the magnitude of the inner product between any basis states  $|e_j\rangle$  and  $|f_k\rangle$  equals the inverse of the dimension  $d$ :

$$|\langle e_j | f_k \rangle|^2 = \frac{1}{d}, \quad \forall j, k \in \{1, \dots, d\}.$$

These bases are unbiased in the following sense: if a system is prepared in a state belonging to one of the bases, then all outcomes of the measurement with respect to the other basis will occur with equal probability. It is known that, for  $d = p^n$  with prime  $p$ , there exists  $d + 1$  MUBs [35].

## 2.6 Quantum property testing

Let  $\mathcal{D}(\mathbb{C}^d)$  denote the set of mixed states in Hilbert space  $\mathbb{C}^d$ , and let a known  $\mathcal{T} \subset \mathcal{D}(\mathbb{C}^d)$  be the working domain of the quantum states. In a standard of property testing scenario, a testing algorithm for a property  $\mathcal{P} \subset \mathcal{T}$  would be an algorithm that, when granted access to independent samples from an unknown quantum state  $\rho \in \mathcal{T}$  as well as an  $\ell_1$  distance parameter of  $0 < \epsilon \leq 1$ , outputs either “Yes” or “No”, with the following guarantees:

- If  $\rho \in \mathcal{P}$ , then it outputs “Yes” with a probability of at least  $\frac{2}{3}$ .
- If  $\rho$  is  $\epsilon$ -far from  $\mathcal{P}$ , then it outputs “No” with a probability of at least  $\frac{2}{3}$ .

Our interest is in designing computational efficient algorithms with the smallest sample complexity (i.e., the smallest number of samples drawn of  $\rho$ ).

Confidence of  $\frac{2}{3}$  is not essential here, it could be replaced by any constant greater than  $\frac{1}{2}$ . This would only change the sample complexity by a multiplicative constant. According to the Chernoff bound, the probability of success becomes  $1 - 2^{-\Omega(k)}$ , after repeating the algorithm  $k$  times.

## 2.7 Tools from earlier work

The following results were established in earlier work, and are used within this paper.

► **Theorem 11** ([11]). *The Pauli group  $\mathcal{P}_k = \{I, X, Y, Z\}^{\otimes n}$  of order  $4^n$  can be divided into  $2^n + 1$  Abelian subgroups with an order of  $2^n$ , say,  $G_0, \dots, G_{2^n}$  such that  $G_i \cap G_j = \{I_2^{\otimes n}\}$  for  $i \neq j$ . Each subgroup can be simultaneously diagonalizable by a corresponding basis. All these  $2^n + 1$  bases form  $2^n + 1$  MUBs.*

► **Theorem 12** ([55, 18]).  *$100 \frac{d}{\epsilon^2}$  copies are sufficient and  $0.15 \frac{d}{\epsilon^2}$  copies are necessary to test whether  $\rho \in \mathcal{D}(\mathbb{C}^d)$  is the maximally mixed state  $\frac{I_d}{d}$  or  $\|\rho - \frac{I_d}{d}\|_1 > \epsilon$  with at least a  $2/3$  probability of success. Generally,  $O(\frac{d}{\epsilon^2})$  copies of  $\rho$  and  $\sigma$  are sufficient to test whether  $\rho = \sigma$  or  $\|\rho - \sigma\|_1 > \epsilon$*

### ■ Algorithm 1 A Mixness Test.

---

**Input:**  $100 \frac{d}{\epsilon^2}$  copies of  $\rho \in \mathcal{D}(\mathbb{C}^d)$

**Output:** “Yes” with a probability of at least  $\frac{2}{3}$  if  $\rho = \frac{I_d}{d}$ ; and “No” with a probability of at least  $\frac{2}{3}$  if  $\|\rho - \frac{I_d}{d}\|_1 > \epsilon$ .

---

### ■ Algorithm 2 A Identity Test with Joint Measurement.

---

**Input:**  $O(\frac{d}{\epsilon^2})$  copies of  $\rho \in \mathcal{D}(\mathbb{C}^d)$  and  $O(\frac{d}{\epsilon^2})$  copies of  $\sigma \in \mathcal{D}(\mathbb{C}^d)$

**Output:** “Yes” with a probability of at least  $\frac{2}{3}$  if  $\rho = \sigma$ ; and “No” with a probability of at least  $\frac{2}{3}$  if  $\|\rho - \sigma\|_1 > \epsilon$ .

---

► **Theorem 13** ([23]). *For  $n$ -dimensional probability distributions of  $p$  and  $q$ ,  $O(\frac{b}{\epsilon^2})$  samples are sufficient to distinguish, with at least a  $\frac{2}{3}$  probability, the cases where  $p = q$  from the cases where  $\|p - q\|_2 > \epsilon$ , where  $b \geq \|p\|_2, \|q\|_2$ .*

### ■ Algorithm 3 An $\ell_2$ norm Identity Test.

---

**Input:**  $O(\frac{b}{\epsilon^2})$  copies of  $p$  and  $O(\frac{b}{\epsilon^2})$  copies of  $q$

**Output:** “Yes” with probability at least  $\frac{2}{3}$  if  $p = q$ , “No” with probability at least  $\frac{2}{3}$  if  $\|p - q\|_2 > \epsilon$ .

---

### 3 Quantum Independence and Technical Lemmata

#### 3.1 Bipartite independence and approximate independence

We say that  $\rho_{1,2} \in \mathcal{D}(\mathbb{C}^d \otimes \mathbb{C}^{d_2})$  is independent if  $\rho_{1,2} = \sigma_1 \otimes \sigma_2$  for some  $\sigma_i \in \mathcal{D}(\mathbb{C}^{d_i})$ . One can directly verify that, if  $\rho_{1,2}$  is independent, then  $\rho = \rho_1 \otimes \rho_2$  with  $\rho_1$  and  $\rho_2$  being the reduced density matrices of  $\rho_{1,2}$ .

We say that  $\rho$  is  $\epsilon$ -independent with respect to the  $\ell_1$  distance if there is an independent state  $\sigma$  such that  $\|\rho - \sigma\|_1 \leq \epsilon$ . We say that  $\rho$  is  $\epsilon$ -far from being independent with respect to the  $\ell_1$  distance if  $\|\rho - \sigma\|_1 > \epsilon$  for any independent state  $\sigma$ .

► **Proposition 14.** *Let  $\rho$  and  $\sigma$  be bipartite states of  $\mathcal{D}(\mathbb{C}^d \otimes \mathbb{C}^{d_2})$ . If  $\|\rho - \sigma\|_1 \leq \epsilon/3$  and  $\sigma$  is independent, then  $\|\rho - \rho_1 \otimes \rho_2\|_1 \leq \epsilon$ .*

► **Lemma 15.**  $\|\rho_1 \otimes \rho_2 - \sigma_1 \otimes \sigma_2\|_1 \leq \|\rho_1 - \sigma_1\|_1 + \|\rho_2 - \sigma_2\|_1$ .

#### 3.2 Multipartite independence and approximate independence

We say that  $\rho \in \mathcal{D}(\mathbb{C}^d \otimes \mathbb{C}^{d_2} \otimes \dots \otimes \mathbb{C}^{d_n})$  is  $n$ -partite independent if  $\rho = \rho_1 \otimes \rho_2 \otimes \dots \otimes \rho_n$ , and that  $\rho$  is  $\epsilon$ -independent with respect to the  $\ell_1$  distance if there is a state  $\sigma$  that is  $m$ -partite independent and  $\|\rho - \sigma\|_1 \leq \epsilon$ . We say that  $\rho$  is  $\epsilon$ -far from being independent with respect to the  $\ell_1$  distance if  $\|\rho - \sigma\|_1 > \epsilon$  for any  $m$ -partite independent state  $\sigma$ .

► **Proposition 16.** *Let  $\rho$  and  $\sigma$  be  $n$ -partite states, if  $\|\rho - \sigma\|_1 \leq \epsilon$ , and  $\sigma$  is  $m$ -partite independent, then  $\|\rho - \rho_1 \otimes \rho_2 \otimes \dots \otimes \rho_n\|_1 \leq (n+1)\epsilon$ .*

► **Lemma 17.**  $\|\rho_1 \otimes \rho_2 \otimes \dots \otimes \rho_n - \sigma_1 \otimes \sigma_2 \otimes \dots \otimes \sigma_n\|_1 \leq \sum_{i=1}^n \|\rho_i - \sigma_i\|_1$ .

Proposition 18 establishes a connection between bipartite independence and multipartite independence. Specifically, it shows that if an  $n$ -partite state is close to bipartite independence in any 1 versus  $n-1$  cut, it is close to being  $n$  partite independent.

► **Proposition 18.** *Let  $\rho$  be an  $n$ -partite states. If for any  $1 \leq i \leq n$ , there exists a state  $\sigma_i^{(i)}$  of party  $i$ , and a state  $\psi_{[n] \setminus \{i\}}$  of parties  $[n] \setminus \{i\}$  such that  $\|\rho - \sigma_i^{(i)} \otimes \psi_{[n] \setminus \{i\}}\|_1 \leq \epsilon$ , then  $\|\rho - \rho_1 \otimes \rho_2 \otimes \dots \otimes \rho_n\|_1 \leq 5n\epsilon$ .*

### 4 Connections between Quantum Property Testing and Distribution Testing

Mutually unbiased bases (MUBs) are used to map the quantum states of  $\mathcal{D}(\mathbb{C}^d)$  into  $d(d+1)$  dimensional probability distributions. Without loss of generality, assume  $d = 2^n$ , and we let the Pauli group  $\mathcal{P}_k = \{I, X, Y, Z\}^{\otimes n}$  be to the order of  $4^n$ . According to Theorem 11, any state  $\rho \in \mathcal{D}(\mathbb{C}^d)$  can be written as

$$\rho = \sum_{P \in \mathcal{P}_n} \eta_P P = \frac{I_d}{d} + \sum_{a=0}^d \sum_{\substack{P \in G_a, \\ P \neq I_d}} \eta_P P = \frac{I_d}{d} + \sum_{i,j} \mu_{i,j} |\beta_{i,j}\rangle \langle \beta_{i,j}|,$$

where  $G_a$  are the Abelian subgroups with an order of  $2^n = d$  such that  $\cup G_a = \mathcal{P}_n$  and  $G_a \cap G_b = \{I_2^{\otimes n}\}$  for  $a \neq b$ . The equation is due to the simultaneous spectrum decomposition of  $G_a$  through the MUBs bases. That is, for  $0 \leq i \neq s \leq d, 1 \leq j, t \leq d$ ,

$$|\langle \beta_{i,j}, \beta_{s,t} \rangle| = \frac{1}{\sqrt{d}}.$$

In addition, it is verifiable that  $\sum_{j=1}^d \mu_{i,j} = 0$  for all  $i$  by the traceless property of  $P \neq I_d$ . Therefore, we can obtain the following constraint on  $\mu_{i,j}$  using  $\sum_{j=1}^d \mu_{i,j} = 0$  for all  $i$ ,

$$\text{Tr } \rho^2 = \text{Tr } \frac{I_d}{d^2} + \sum_{i,j,s,t} \mu_{i,j} \mu_{s,t} |\langle \beta_{i,j}, \beta_{s,t} \rangle|^2 = \frac{1}{d} + \sum_{i,j} \mu_{i,j}^2 \leq 1.$$

$\mathcal{M} = \{M_{ij} = \frac{|\beta_{i,j}\rangle\langle\beta_{i,j}|}{d+1} : 0 \leq i \leq d, 1 \leq j \leq d\}$  can be used to map the  $d$ -dimensional quantum state  $\rho$  into a  $d(d+1)$  dimensional probabilistic distribution. The corresponding probability distribution  $p = (p(0,1), \dots, p(d,d))$  satisfies

$$p(i,j) = \frac{\text{Tr}(\rho |\beta_{i,j}\rangle\langle\beta_{i,j}|)}{d+1} = \frac{\mu_{i,j} + \frac{1}{d}}{d+1},$$

note that other terms are orthogonal or cancel out due to the property of MUBs and the equations  $\sum_{j=1}^d \mu_{i,j} = 0$  for all  $i$ .

Then the  $\ell_2$  norm of  $p$  can be bounded with

$$\frac{\sqrt{\sum_{i,j} (\mu_{i,j} + \frac{1}{d})^2}}{d+1} = \frac{\sqrt{\sum_{i,j} \mu_{i,j}^2 + \frac{d(d+1)}{d^2} + \frac{2 \sum_{i,j} \mu_{i,j}}{d}}}{d+1} = \frac{\sqrt{\sum_{i,j} \mu_{i,j}^2 + \frac{d+1}{d}}}{d+1} \leq \frac{\sqrt{2}}{d+1}.$$

More importantly, this map preserves the  $\ell_2$  distance, in the sense that the  $\ell_2$  distance between the image probability distributions is exactly the same as the  $\ell_2$  distance between the pre-image quantum states with a scaling of  $\frac{1}{d+1}$ .

For any two states  $\rho = \frac{I_d}{d} + \sum_{i,j} \mu_{i,j} |\beta_{i,j}\rangle\langle\beta_{i,j}|$  and  $\sigma = \frac{I_d}{d} + \sum_{i,j} \nu_{i,j} |\beta_{i,j}\rangle\langle\beta_{i,j}|$ , we have that

$$\|\rho - \sigma\|_2 = \left\| \sum_{i,j} (\mu_{i,j} - \nu_{i,j}) |\beta_{i,j}\rangle\langle\beta_{i,j}| \right\|_2 = \sqrt{\sum_{i,j} (\mu_{i,j} - \nu_{i,j})^2},$$

where the other terms are orthogonal or cancel out due to the property of MUBs and the equation  $\sum_{j=1}^d \mu_{i,j} = 0$  for all  $i$ .

Using the measurement  $\mathcal{M}$ , the corresponding probability distributions can be obtained:  $p = (p(0,1), \dots, p(d,d))$  and  $q = (q(0,1), \dots, q(d,d))$  with

$$p(i,j) = \frac{\text{Tr}(\rho |\beta_{i,j}\rangle\langle\beta_{i,j}|)}{d+1} = \frac{\mu_{i,j} + \frac{1}{d}}{d+1}, \quad q(i,j) = \frac{\text{Tr}(\sigma |\beta_{i,j}\rangle\langle\beta_{i,j}|)}{d+1} = \frac{\nu_{i,j} + \frac{1}{d}}{d+1}.$$

The following equality proves Lemma 2.  $\|p - q\|_2 = \frac{\sqrt{\sum_{i,j} (\mu_{i,j} - \nu_{i,j})^2}}{d+1} = \frac{\|\rho - \sigma\|_2}{d+1}.$

## 5 Quantum State Certification

The connections developed in Section 4, together with the  $\ell_2$ -identity tester of probability distributions provided in [23], also make efficient identity testing of quantum states possible.

**Proof of Theorem 3.** First map the state into probability distributions, say  $p$  and  $q$ , through independent measurement with Theorem 2, and follow by executing Algorithm 4.

■ **Algorithm 4** A Identity Test with Independent Measurement.

---

**Input:**  $O(\frac{d^2}{\epsilon^2})$  copies of  $\rho \in \mathcal{D}(\mathbb{C}^d)$  and  $O(\frac{d}{\epsilon^2})$  copies of  $\sigma \in \mathcal{D}(\mathbb{C}^d)$   
**Output:** “Yes” with a probability of at least  $\frac{2}{3}$  if  $\rho = \sigma$ ; and “No” with a probability of at least  $\frac{2}{3}$  if  $\|\rho - \sigma\|_1 > \epsilon$ .  
1 Run Algorithm 3 to distinguish between  $p = q$  and  $\|p - q\|_2 \geq \frac{\epsilon}{\sqrt{d(d+1)}}$ ;  
/\*  $p$  and  $q$  are the probability distributions obtained by measuring  $\rho$   
and  $\sigma$  through the independent measurement with Theorem 2,  
respectively. \*/

---

According to  $\|p - q\|_2 = \frac{\|\rho - \sigma\|_2}{\sqrt{d+1}}$ , we only need to distinguish cases where  $p = q$  from cases where  $\|p - q\|_2 \geq \frac{\|\rho - \sigma\|_1}{\sqrt{d(d+1)}} \geq \frac{\epsilon}{\sqrt{d(d+1)}}$ . Choosing  $b = \frac{\sqrt{2}}{d+1} \geq \|p\|_2, \|q\|_2$  and invoking Theorem 13, we have

$$O\left(\frac{b}{\left(\frac{\epsilon}{\sqrt{d(d+1)}}\right)^2}\right) = O\left(\frac{d^2}{\epsilon^2}\right)$$

which is a sufficient number of copies. ◀

According to [44], the sample complexity for tomography is  $\rho \in \mathcal{D}(\mathbb{C}^d)$  is  $\Theta(\frac{d^3}{\epsilon^2})$ , which makes Algorithm 4 a better choice for identity testing after tomography.

As mentioned in the introduction, Algorithm 4 should be significantly easier to implement because it does not demand noiseless, universal quantum computation with an *exponential* number of qubits.

## 6 Independence Testing

The goal of independence testing is to determine whether a fixed multipartite state  $\rho$  is independent, i.e., in tensor product form, or far from being independent. Hence, in this section, we outline a series of testing algorithms and almost matching lower bounds in joint measurement setting, and independent measurement setting.

We start with an algorithm for the bipartite case of Theorem 5.

■ **Algorithm 5** A Bipartite Independence Testing with Joint Measurement.

---

**Input:**  $n = O(\frac{d_1 d_2}{\epsilon^2})$  copies of  $\rho \in \mathcal{D}(\mathbb{C}^{d_1} \otimes \mathbb{C}^{d_2})$   
**Output:** “Yes” with a probability of at least  $\frac{2}{3}$  if  $\rho$  is independent; and “No” with a probability of at least  $\frac{2}{3}$  if  $\|\rho - \sigma\|_1 > \epsilon$  for any independent  $\sigma$ .

- 1 Use  $\frac{n}{3}$  copies of  $\rho$  to generate  $\rho_1$ ;  
 /\* Trace out system 2 \*/
- 2 Use  $\frac{n}{3}$  copies of  $\rho$  to generate  $\rho_2$ ;  
 /\* Trace out system 1 \*/
- 3 Run Algorithm 2 on  $\frac{n}{3}$  copies of  $\rho$  and  $\frac{n}{3}$  copies of  $\rho_1 \otimes \rho_2$  with the parameter  $\epsilon/3$ ;

---

**Proof.** The correctness of this algorithm accords with Theorem 1 by note that

- If  $\rho$  is independent, then  $\rho = \rho_1 \otimes \rho_2$ , and this algorithm will output “Yes” with high probability.
- If  $\|\rho - \sigma\|_1 > \epsilon$  for any independent  $\sigma$ , then  $\|\rho - \rho_1 \otimes \rho_2\|_1 > \epsilon/3$  by Proposition 14, and this algorithm will output “No” with high probability.

We can derive an independent measurement tester by replacing the identity tester in Algorithm 2 with Algorithm 4. From a similar analysis to the above,  $O(\frac{d_1^2 d_2^2}{\epsilon^2})$  is a sufficient number of copies. ◀

The obvious generalization of the bipartite independence testing to  $m$ -partite would work using bipartite independence in any  $n - 1$  parties versus 1 party. Our goal is to test independence in this scenario with an accuracy of  $O(\frac{\epsilon}{n})$  and at least a  $1 - \frac{1}{n^2}$  probability of success. The correctness of the algorithm follows from Proposition 18, and the generalization incurs an  $O(n^3 \log n)$  factor. For constant  $n$ ,  $O(n^3 \log n)$  is still constant. Thus, the complexity of the different algorithm variants would be  $O(\frac{\prod_{i=1}^n d_i}{\epsilon^2})$  with joint measurement, and  $O(\frac{\prod_{i=1}^n d_i^2}{\epsilon^2})$  with independent measurement. With a super-constant  $n$ , algorithms could be built that achieve the same complexity using Diakonikolas and Kane’s [31] recursion idea coupled with our previous bipartite independence tester.

We only prove the lower bound part of Theorem 5 for bipartite systems here. The general version can be proved similarly. In cases where  $d_1$  and  $d_2$  are both very large, the bound is derived from the mixness test of Theorem 12 in [55], where the constant 2000 comes from the upper and lower bound of the constant in that theorem. To deal with “unbalanced” cases where only  $d_1$  or  $d_2$  is small—here, let us say  $d_2$ —we split the  $d_1$  system into many systems of dimension  $d_2$ , which transforms the original unbalance of a bipartite problem into a problem of “balanced” multipartite independence testing. Then, we use Proposition 18.

**Proof.** First, note that it suffices to consider cases where  $d_1 d_2$  are sufficiently large. To show the lower bound for a general  $d_1$  and  $d_2$ , assume there is an algorithm, Algorithm A, that uses  $f(d_1, d_2, \epsilon)$  copies to decide whether a given  $\rho \in \mathcal{D}(\mathbb{C}^{d_1} \otimes \mathbb{C}^{d_2})$  is independent or  $\epsilon$ -far from being independent with at least a  $2/3$  probability of successful. By using Algorithm A as an oracle, the following algorithm can distinguish cases where  $\rho = \frac{I_{d_1}}{d_1} \otimes \frac{I_{d_2}}{d_2}$  from cases where  $\|\rho - \frac{I_{d_1}}{d_1} \otimes \frac{I_{d_2}}{d_2}\|_1 > \epsilon$  for any  $t > 1$ .

To see this algorithm to succeed at detecting whether  $\rho$  is maximally mixed with high probability, note that: If  $\rho = \frac{I_{d_1}}{d_1} \otimes \frac{I_{d_2}}{d_2}$ , in Line 1, the algorithm will output  $\rho_1 = \frac{I_{d_1}}{d_1}$  with a probability of at least  $\frac{20}{27}$ ; in Line 5, the algorithm will output  $\rho_2 = \frac{I_{d_2}}{d_2}$  with a probability of at least  $\frac{27}{28}$ ; in Line 9,  $\rho$  will be independent with a probability of at least  $\frac{28}{30}$ . Overall, Algorithm 6 will output “Yes” with a probability of at least  $\frac{2}{3}$ .



## 11:14 Quantum Identity Testing and Independence Testing

If  $\|\rho - \frac{I_{d_1}}{d_1} \otimes \frac{I_{d_2}}{d_2}\|_1 > \epsilon$ , then one of the following three statements will be true:  $\rho_1$  is  $\epsilon/t$ -far from  $\frac{I_{d_1}}{d_1}$ ; or  $\rho_2$  is  $\frac{(t-1)\epsilon}{4t}$ -far from  $\frac{I_{d_2}}{d_2}$ ; or  $\rho$  is  $\frac{(t-1)\epsilon}{4t}$ -far from being independent. Otherwise, assume that there exists an  $\sigma_1$  and an  $\sigma_2$ , such that  $\|\rho - \sigma_1 \otimes \sigma_2\|_1 < \frac{(t-1)\epsilon}{4t}$ ,  $\|\rho_1 - \frac{I_{d_1}}{d_1}\|_1 < \frac{\epsilon}{t}$  and  $\|\rho_2 - \frac{I_{d_2}}{d_2}\|_1 < \frac{(t-1)\epsilon}{4t}$ . According to Proposition 14, we have  $\|\rho - \rho_1 \otimes \rho_2\|_1 < \frac{3(t-1)\epsilon}{4t}$ . Then by the triangle inequality and Lemma 15, we have

$$\|\rho - \frac{I_{d_1}}{d_1} \otimes \frac{I_{d_2}}{d_2}\|_1 \leq \|\rho - \rho_1 \otimes \rho_2\|_1 + \|\frac{I_{d_1}}{d_1} \otimes \frac{I_{d_2}}{d_2} - \rho_1 \otimes \rho_2\|_1 < \epsilon.$$

Contradiction! Therefore, in this case, the algorithm outputs “No” with a probability of at least  $\min\{\frac{20}{27}, \frac{27}{28}, \frac{28}{30}\} > \frac{2}{3}$ .

■ **Algorithm 6** A Bipartite Identity test A for a maximally mixed state.

---

**Input:**  $n = 100f(d_1, d_2, \frac{(t-1)\epsilon}{4t}) + 300t^2 \frac{d_1}{\epsilon^2} + \Theta(\frac{d_2}{t^2(t-1)^2\epsilon^2})$  copies of  $\rho \in \mathcal{D}(\mathbb{C}^{d_1} \otimes \mathbb{C}^{d_2})$

**Output:** “Yes” with a probability of at least  $\frac{2}{3}$  if  $\rho = \frac{I_{d_1}}{d_1} \otimes \frac{I_{d_2}}{d_2}$ ; and “No” with a probability of at least  $\frac{2}{3}$  if  $\|\rho - \frac{I_{d_1}}{d_1} \otimes \frac{I_{d_2}}{d_2}\|_1 > \epsilon$ .

- 1 Repeat Algorithm 1, with  $100t^2 \frac{d_1}{\epsilon^2}$  copies of  $\rho$ , three times to test whether  $\rho_1 = \frac{I_{d_1}}{d_1}$  or  $\|\rho_1 - \frac{I_{d_1}}{d_1}\|_1 > \epsilon/t$  with at least a  $\frac{20}{27}$  probability of success;
- 2 **if** “No” **then**
- 3     Return “No”;
- 4 **else**
- 5     Employ Algorithm 1 with  $\Theta(\frac{t^2 d_2}{(t-1)^2 \epsilon^2})$  copies of  $\rho$  to test whether  $\rho_2 = \frac{I_{d_2}}{d_2}$  or  $\|\rho_2 - \frac{I_{d_2}}{d_2}\|_1 > \frac{(t-1)\epsilon}{4t}$  with at least a  $\frac{27}{28}$  probability of success;
- 6     **if** “No” **then**
- 7         Return “No”;
- 8     **else**
- 9         Run Algorithm A 100 times to test whether  $\rho$  is independent or is  $\frac{(t-1)\epsilon}{4t}$ -far from being independent with at least a  $\frac{28}{30}$  probability of success;
- 10        **if** “Yes” **then**
- 11            Return “Yes”;
- 12        **else**
- 13            Return “No”;

---

This algorithm uses  $n = 100f(d_1, d_2, \frac{(t-1)\epsilon}{4t}) + 300t^2 \frac{d_1}{\epsilon^2} + \Theta(\frac{t^2 d_2}{(t-1)^2 \epsilon^2})$  copies of  $\rho$ . Invoking Theorem 12, we know that  $0.15 \frac{d_1 d_2}{\epsilon^2}$  copies are necessary to test, with at least a  $2/3$  probability of success, whether  $\rho$  is the maximally mixed state or whether it is  $\epsilon$ -far.

We must have

$$100f(d_1, d_2, \frac{(t-1)\epsilon}{4t}) + 300t^2 \frac{d_1}{\epsilon^2} + \Theta(t^2 \frac{d_2}{(t-1)^2 \epsilon^2}) \geq 0.15 \frac{d_1 d_2}{\epsilon^2}.$$

If  $d_1$  and  $d_2$  are both sufficiently large, we can choose a constant  $t$  such that  $300t^2 \frac{d_1}{\epsilon^2} + \Theta(t^2 \frac{d_2}{(t-1)^2 \epsilon^2}) = o(\frac{d_1 d_2}{\epsilon^2})$ , which implies

$$f(d_1, d_2, \epsilon) \geq \Omega(\frac{16t^2 d_1 d_2}{(t-1)^2 \epsilon^2}) = \Omega(\frac{d_1 d_2}{\epsilon^2}).$$

If  $d_1$  is sufficiently large and  $d_2$  is not sufficiently large but  $d_2 > 2000$ , we can choose  $t = \sqrt{\frac{2000.5}{2000}}$ , then

$$f(d_1, d_2, c\epsilon) \geq 0.15 \frac{d_1 d_2}{\epsilon^2} - 300t^2 \frac{d_1}{\epsilon^2} + \Omega(t^2 \frac{d_2}{(t-1)^2 \epsilon^2}) = \Omega(\frac{d_1}{\epsilon^2}) = \Omega(\frac{d_1 d_2}{\epsilon^2}),$$

with the constant  $c = \frac{t-1}{4t}$ . Thus, for  $d_2 > 2000$ ,

$$f(d_1, d_2, \epsilon) \geq \Omega(\frac{d_1 d_2}{\epsilon^2}).$$

The above technique does not work with a small  $d_2$ , because the number of copies required to test a  $d_1$  system  $300t^2 \frac{d_1}{\epsilon^2}$  and the number of copies required to test a total system of  $0.15 \frac{d_1 d_2}{\epsilon^2}$  are of the same order.

To deal with this unbalanced case, we developed a dimension splitting technique that transforms bipartite independence into  $k$ -partite independence. First observe that the sample complexity for independence testing in  $\mathcal{D}(\mathbb{C}^{d_1} \otimes \mathbb{C}^{d_2})$  is no less than the sample complexity for an independence test of  $\mathcal{D}(\mathbb{C}^d \otimes \mathbb{C}^2)$  for  $d = 2^{\lceil \log d_1 \rceil} \leq d_1$ . Therefore, without loss of generality, assume that  $d_2 = 2$  and  $d_1 = 2^k$  instead of  $d_2 \leq 2000$ , and that  $d_1$  is sufficiently large.

We still assume that there is an Algorithm A that uses  $f(2^k, 2, \epsilon)$  copies to decide, with at least a  $2/3$  probability of success, whether a given  $\rho \in \mathcal{D}(\mathbb{C}^{2^k \times 2^k} \otimes \mathbb{C}^{2 \times 2})$  is independent or  $\epsilon$ -far from independent in the  $2^k$  and 2 bipartitions. Any such  $\rho$  can be regarded as a  $k+1$  qubit state, and the qubit systems will be labeled as  $S = \{1, 2, \dots, k, k+1\}$ .  $\rho_i$  denotes the reduced density matrix of the  $i$ -th qubit of  $\rho$ . Algorithm A is a bipartite independence tester for a  $k+1$  qubit system in the bipartition of  $k$  qubits and 1 qubit. In the following, Algorithm A is applied as a black box to the bipartition  $i$  and  $S \setminus \{i\}$  for any  $i$  to test the identity of  $\rho$  and  $\frac{I_{d_1}}{d_1} \otimes \frac{I_{d_2}}{d_2}$ .

■ **Algorithm 7** A Bipartite Identity Test B for a maximally mixed state.

**Input:**  $n = \Theta[(k+1) \log k f(2^k, 2, \frac{\epsilon}{6(k+1)})] + \Theta[(k+1) \log k \frac{(k+1)^2}{\epsilon^2}]$  copies of  $\rho$ .

**Output:** “Yes” with a probability of at least  $\frac{2}{3}$  if  $\rho = \otimes_{i=1}^{k+1} \frac{I_2}{2}$ ; and “No” with a probability of at least  $\frac{2}{3}$  if  $\|\rho - \otimes_{i=1}^{k+1} \frac{I_2}{2}\|_1 > \epsilon$ .

```

1 for  $i \leftarrow 1$  to  $k+1$  do
2   Repeat Algorithm 1, with  $\Theta(\frac{(k+1)^2}{\epsilon^2})$  copies of  $\rho$  each time,  $\Theta(\log k)$  times to test
   whether  $\rho_i = \frac{I_2}{2}$  or  $\|\rho_i - \frac{I_2}{2}\|_1 > \frac{\epsilon}{6(k+1)}$  at least a  $1 - \frac{1}{k^2}$  probability of success;
3   if No then
4     Return “No”;
5   else
6     Run Algorithm A  $\Theta(\log k)$  times, with  $f(2^k, 2, \frac{\epsilon}{6(k+1)})$  copies each time, to
     test whether  $\rho$  is independent or  $\frac{\epsilon}{6(k+1)}$ -far from being independent in the
     bipartition  $\{i\}$  and  $S \setminus \{i\}$  with at least a  $1 - \frac{1}{k^2}$  probability of success;
7     if No then
8       Return “No”;
9 Return “Yes”;
```

To see this algorithm succeed in detecting whether  $\rho$  is maximally mixed with high probability, we note that

## 11:16 Quantum Identity Testing and Independence Testing

If  $\rho = \frac{I_{d_1}}{d_1} \otimes \frac{I_{d_2}}{d_2}$ , then  $\rho_i = \frac{I_2}{2}$  when  $\rho$  is regarded as a  $k+1$  qubit state. It is independent in any bipartition  $\{i\}$  and  $S \setminus \{i\}$ . For each  $i$ , the passing probability of the test  $\rho_i = \frac{I_2}{2}$  is at least  $1 - \frac{1}{k^2}$ . For each  $i$ , the passing probability of the independence test in the bipartition  $\{i\}$  and  $S \setminus \{i\}$  is at least  $1 - \frac{1}{k^2}$ . In total, Algorithm 7 will accept with a probability of at least  $(1 - \frac{1}{k^2})^{O(k)} = 1 - o(1) > \frac{2}{3}$ .

If  $\|\rho - \frac{I_{d_1}}{d_1} \otimes \frac{I_{d_2}}{d_2}\|_1 > \epsilon$ , at least one of the following two statements is true:

- $\|\rho_i - \frac{I_2}{2}\|_1 > \frac{\epsilon}{6(k+1)}$  for some  $1 \leq i \leq k+1$ ; and/or
- $\rho$  is  $\frac{\epsilon}{6(k+1)}$ -far from independent in the bipartition  $\{i\}$  and  $S \setminus \{i\}$  for some  $1 \leq i \leq k+1$ .

Otherwise,  $\|\rho_i - \frac{I_2}{2}\|_1 \leq \frac{\epsilon}{6(k+1)}$  and  $\rho$  is  $\frac{\epsilon}{6(k+1)}$  close to being independent in the bipartition  $\{i\}$  and  $S \setminus \{i\}$  for all  $1 \leq i \leq k+1$ .

According to Proposition 18, we have

$$\|\rho - \rho_1 \otimes \rho_2 \otimes \cdots \otimes \rho_{k+1}\|_1 \leq 5(k+1) \frac{\epsilon}{6(k+1)} = \frac{5\epsilon}{6}.$$

By Lemma 17, we have

$$\begin{aligned} & \|\rho - \frac{I_{d_1}}{d_1} \otimes \frac{I_{d_2}}{d_2}\|_1 \\ &= \|\rho - \frac{I_2}{2} \otimes \frac{I_2}{2} \otimes \cdots \otimes \frac{I_2}{2}\|_1 \\ &\leq \|\rho - \rho_1 \otimes \rho_2 \otimes \cdots \otimes \rho_{k+1}\|_1 + \|\frac{I_2}{2} \otimes \frac{I_2}{2} \otimes \cdots \otimes \frac{I_2}{2} - \rho_1 \otimes \rho_2 \otimes \cdots \otimes \rho_{k+1}\|_1 \\ &\leq \frac{5\epsilon}{6} + \sum_{i=1}^{k+1} \|\rho_i - \frac{I_2}{2}\|_1 \\ &\leq \epsilon. \end{aligned}$$

Contradiction! Therefore, the algorithm outputs “No” with a probability of at least  $1 - \frac{1}{k^2} > \frac{2}{3}$  in this case.

Invoking Theorem 12, we know that  $\Theta(\frac{d_1 d_2}{\epsilon^2}) = \Theta(\frac{2^{k+1}}{\epsilon^2})$  copies are necessary to test whether  $\rho$  is a maximally mixed state or  $\epsilon$ -far with at least a  $2/3$  probability of success. Algorithm 7 uses  $\Theta[(k+1) \log k f(2^k, 2, \frac{\epsilon}{6(k+1)})] + \Theta[(k+1) \log k \frac{(k+1)^2}{\epsilon^2}]$  copies of  $\rho$ . We must have

$$\begin{aligned} & \Theta[(k+1) \log k f(2^k, 2, \frac{\epsilon}{6(k+1)})] + \Theta[(k+1) \log k \frac{(k+1)^2}{\epsilon^2}] \geq \Theta(\frac{2^{k+1}}{\epsilon^2}) \\ \Rightarrow & f(2^k, 2, \frac{\epsilon}{6(k+1)}) \geq \Theta(\frac{2^k}{k \log k \epsilon^2}) \\ \Rightarrow & f(2^k, 2, \epsilon) \geq \Theta(\frac{2^k}{k^3 \log k \epsilon^2}) \\ \Rightarrow & f(d_1, d_2, \epsilon) = \Omega(\frac{d_1}{\log^3 d_1 \log \log d_1 \epsilon^2}) = \Omega(\frac{d_1 d_2}{\log^3 d_2 \log \log d_1 \epsilon^2}) \end{aligned}$$

That is, if  $d_2$  is a small constant,  $\Omega(\frac{d_1 d_2}{\epsilon^2 \log^3 d_1 \log \log d_1})$  copies are necessary to test the independence of  $\rho \in \mathcal{D}(\mathbb{C}^{d_1} \otimes \mathbb{C}^{d_2})$ . ◀

## 7 Discussion

If we can use random measurements, a fewer number of copies are needed for quantum identity testing and independence testing.

In [19], it is proved that using non-adaptive independent measurements, to test whether a quantum state  $\rho \in \mathcal{D}(\mathbb{C}^d)$  is equal to or  $\epsilon$ -far in trace distance from the maximally mixed state,  $\Omega(\frac{d^{3/2}}{\epsilon^2})$ , and this complexity can be achieved via Haar-random orthogonal POVMs. The measurement can be implemented by randomly choosing a unitary  $U$  applied on  $\rho$  and measuring  $U\rho U^\dagger$  in computational basis many times. The last step is to test whether the resulting  $d$ -dimensional probability distribution  $p_U$  is equal to or  $\frac{\epsilon}{\sqrt{d}}$ -far from uniform distribution  $u$ . The correctness of this algorithm comes from the concentration of measure and

$$\mathbb{E}_U \|p_U - u\|_2^2 = \frac{\|\rho - \frac{I}{d}\|_2^2}{d+1}.$$

This method can be used to the general quantum identity testing problem: Randomly choosing a unitary  $U$  applied on  $\rho$  and  $\sigma$  respectively, then measuring  $U\rho U^\dagger$  and  $U\sigma U^\dagger$  in computational basis many times. The last step is to test whether the resulting  $d$ -dimensional probability distributions  $p_U$  and  $q_U$  are equal or  $\frac{\epsilon}{\sqrt{d}}$ -far. One can verify

$$\mathbb{E}_U \|p_U - q_U\|_2^2 = \frac{\|\rho - \sigma\|_2^2}{d+1},$$

and

$$\mathbb{E}_U \|p_U\|_2^2, \|q_U\|_2^2 \leq O\left(\frac{1}{d+1}\right).$$

Using concentration of measure, we know that  $\|p_U - q_U\|_2^2 \geq \frac{\|\rho - \sigma\|_2^2}{2d+1}$  and  $\|p_U\|_2^2, \|q_U\|_2^2 \leq O(\frac{1}{d+1})$  are valid with high probability. The rest is to run Algorithm 3 from [23] with  $O(\frac{d^{3/2}}{\epsilon^2})$  samples. We can conclude that

► **Theorem 19.** *The sample complexity of quantum identity testing is  $\Theta(\frac{d^{3/2}}{\epsilon^2})$  for non-adaptive independent measurements.*

This continuous randomness can be discretized by randomly choosing the MUB basis presented in Section 4.

Plug in our method of quantum independence testing, we know that

► **Theorem 20.** *The sample complexity of quantum independence testing of  $\mathcal{D}(\mathbb{C}^{d_1} \otimes \mathbb{C}^{d_2} \otimes \dots \otimes \mathbb{C}^{d_m})$  is  $\tilde{\Theta}(\frac{\prod_{i=1}^m d_i^{3/2}}{\epsilon^2})$  for non-adaptive independent measurements.*

---

## References

- 1 S. Aaronson. Shadow tomography of quantum states. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2018, pages 325–338, 2018.
- 2 S. Aaronson, X. Chen, E. Hazan, S. Kale, and A. Nayak. Online learning of quantum states. In *Advances in Neural Information Processing Systems 31*, pages 8962–8972, 2018.
- 3 S. Aaronson and G. Rothblum. Gentle measurement of quantum states and differential privacy. In *Proceedings of the 51th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2019, 2019.
- 4 Scott Aaronson. The learnability of quantum states. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 463(2088):3089–3114, September 2007.
- 5 J. Acharya, H. Das, A. Jafarpour, A. Orlitsky, and S. Pan. Competitive closeness testing. In *Proceedings of the 24th Annual Conference on Learning Theory*, volume 19, pages 47–68, 2011.
- 6 J. Acharya, C. Daskalakis, and G. Kamath. Optimal testing for properties of distributions. In *Advances in Neural Information Processing Systems 28*, pages 3591–3599, 2015.

- 7 J. Acharya, I. Diakonikolas, J. Li, and L. Schmidt. Sample-optimal density estimation in nearly-linear time. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '17, pages 1278–1289, 2017.
- 8 J. Acharya, I. Issa, N. Shende, and A. B. Wagner. Measuring quantum entropy. *1711.00814*, 2017.
- 9 Anurag Anshu, Dave Touchette, Penghui Yao, and Nengkun Yu. Exponential separation of quantum communication and classical information. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2017, pages 277–288, New York, NY, USA, 2017. ACM. doi:10.1145/3055399.3055401.
- 10 Leandro Aolita, Christian Gogolin, Martin Kliesch, and Jens Eisert. Reliable quantum certification of photonic state preparations. *Nature Communications*, 6, 2015.
- 11 Bandyopadhyay, Boykin, Roychowdhury, and Vatan. A new proof for the existence of mutually unbiased bases. *Algorithmica*, 34(4):512–528, 2002.
- 12 T. Batu, S. Dasgupta, R. Kumar, and R. Rubinfeld. The complexity of approximating entropy. In *Proceedings of the Thirty-fourth Annual ACM Symposium on Theory of Computing*, STOC '02, pages 678–687, 2002.
- 13 T. Batu, L. Fortnow, E. Fischer, R. Kumar, R. Rubinfeld, and P. White. Testing random variables for independence and identity. In *Proceedings of the 42Nd IEEE Symposium on Foundations of Computer Science*, FOCS '01, pages 442–451, 2001.
- 14 T. Batu, L. Fortnow, R. Rubinfeld, W. D. Smith, and P. White. Testing that distributions are close. In *Proceedings 41st Annual Symposium on Foundations of Computer Science*, FOCS'00, pages 259–269, 2000.
- 15 T. Batu, R. Kumar, and R. Rubinfeld. Sublinear algorithms for testing monotone and unimodal distributions. In *Proceedings of the Thirty-sixth Annual ACM Symposium on Theory of Computing*, STOC '04, pages 381–390, 2004.
- 16 S. Bravyi, A. W. Harrow, and A. Hassidim. Quantum algorithms for testing properties of distributions. *IEEE Transactions on Information Theory*, 57(6):3971–3981, 2011.
- 17 C. Bădescu and R. O'Donnell. Lower bounds for testing complete positivity and quantum separability. In *14th Latin American Theoretical Informatics Symposium*, 2020.
- 18 C. Bădescu, R. O'Donnell, and J. Wright. Quantum state certification. In *Proceedings of the Forty-Ninth Annual ACM Symposium on Theory of Computing*, STOC '19, 2019.
- 19 Sébastien Bubeck, Sitan Chen, and Jerry Li. Entanglement is necessary for optimal quantum property testing. In *FOCS 2020*, April 2020.
- 20 C. L. Canonne. A survey on distribution testing: Your data is big. but is it blue? *Electronic Colloquium on Computational Complexity (ECCC)*, 22:63, 2015.
- 21 C. L. Canonne, I. Diakonikolas, T. Gouleakis, and R. Rubinfeld. Testing shape restrictions of discrete distributions. *Theory of Computing Systems*, 62(1):4–62, 2018.
- 22 C. L. Canonne, I. Diakonikolas, D. M. Kane, and A. Stewart. Testing conditional independence of discrete distributions. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2018, pages 735–748, 2018.
- 23 S. Chan, I. Diakonikolas, G. Valiant, and P. Valiant. Optimal algorithms for testing closeness of discrete distributions. In *Proceedings of the Twenty-fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '14, pages 1193–1203, 2014.
- 24 Y. Cheng, I. Diakonikolas, and R. Ge. High-dimensional robust mean estimation in nearly-linear time. In *Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms*, 2019.
- 25 Marcus P. da Silva, Olivier Landon-Cardinal, and David Poulin. Practical characterization of quantum devices without tomography. *Phys. Rev. Lett.*, 107:210404, 2011.
- 26 C. Daskalakis, I. Diakonikolas, R. A. Servedio, G. Valiant, and P. Valiant. Testing k-modal distributions: Optimal algorithms via reductions. In *Proceedings of the Twenty-fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '13, pages 1833–1852, 2013.

- 27 C. Daskalakis, N. Dikkala, and G. Kamath. Testing ising models. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '18, pages 1989–2007, 2018.
- 28 C. Daskalakis and Q. Pan. Square hellinger subadditivity for bayesian networks and its applications to identity testing. In *Proceedings of the 2017 Conference on Learning Theory*, volume 65, pages 697–703, 2017.
- 29 L. Devroye and G. Lugosi. *Combinatorial Methods in Density Estimation*. Springer, 2001.
- 30 I. Diakonikolas, G. Kamath, D. M. Kane, J. Li, A. Moitra, and A. Stewart. Robust estimators in high dimensions without the computational intractability. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 655–664, 2016.
- 31 I. Diakonikolas and D. Kane. A new approach for testing properties of discrete distributions. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 685–694, 2016.
- 32 I. Diakonikolas, D. M. Kane, and V. Nikishkin. Optimal algorithms and lower bounds for testing closeness of structured distributions. In *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, pages 1183–1202, 2015.
- 33 I. Diakonikolas, D. M. Kane, and V. Nikishkin. Near-Optimal Closeness Testing of Discrete Histogram Distributions. In *44th International Colloquium on Automata, Languages, and Programming (ICALP 2017)*, volume 80, pages 8:1–8:15, 2017.
- 34 I. Diakonikolas, D. M. Kane, and A. Stewart. Sharp bounds for generalized uniformity testing. In *Advances in Neural Information Processing Systems 31*, pages 6201–6210, 2018.
- 35 T. Durt, B. Englert, I. Bengtsson, and Zyczkowski K. On mutually unbiased bases. *International Journal of Quantum Information*, pages 535–640, 2010.
- 36 S. T. Flammia, D. Gross, Y. Liu, and J. Eisert. Quantum tomography via compressed sensing: Error bounds, sample complexity, and efficient estimators. *New J. Phys.*, 14:095022, 2012.
- 37 Steven T. Flammia and Yi-Kai Liu. Direct fidelity estimation from few pauli measurements. *Phys. Rev. Lett.*, 106:230501, 2011.
- 38 András Gilyén and Tongyang Li. Distributional Property Testing in a Quantum World. In *11th Innovations in Theoretical Computer Science Conference (ITCS 2020)*, volume 151, pages 25:1–25:19, 2020.
- 39 O. Goldreich. *Introduction to Property Testing*. Cambridge University Press, 2017.
- 40 O. Goldreich and D. Ron. *On Testing Expansion in Bounded-Degree Graphs*, volume 6650 of *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation, Lecture Notes in Computer Science*. Springer, 2000.
- 41 Oded Goldreich, Shari Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *J. ACM*, 45(4):653–750, July 1998.
- 42 D. Gross, Y. Liu, S. T. Flammia, S. Becker, and J. Eisert. Quantum state tomography via compressed sensing. *Phys. Rev. Lett.*, 105(150401), 2010.
- 43 D. Gross, S. Nezami, and M. Walter. Schur-weyl duality for the clifford group with applications: Property testing, a robust hudson theorem, and de finetti representations. *arXiv*, 2017. [arXiv:1712.08628](https://arxiv.org/abs/1712.08628).
- 44 J. Haah, A. W. Harrow, Z. Ji, X. Wu, , and N. Yu. Sample-optimal tomography of quantum states. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing*, STOC '16, pages 913–925, 2016.
- 45 Aram W. Harrow and Ashley Montanaro. Testing product states, quantum merlin-arthur games and tensor optimization. *J. ACM*, 60(1):3:1–3:43, 2013.
- 46 P. Indyk, R. Levi, and R. Rubinfeld. Approximating and testing k-histogram distributions in sub-linear time. In *Proceedings of the 31st ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, PODS '12, pages 15–22, 2012.
- 47 R. Jain, J. Radhakrishnan, and Sen P. A lower bound for the bounded round quantum communication complexity of set disjointness. In *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.*, pages 220–229, October 2003. doi:10.1109/SFCS.2003.1238196.


- 48 J. Jiao, K. Venkat, Y. Han, and T. Weissman. Minimax estimation of functionals of discrete distributions. *IEEE Transactions on Information Theory*, 61(5):2835–2885, 2015.
- 49 R. Kueng, H. Rauhut, and U. Terstiege. Low rank matrix recovery from rank one measurements. *Applied and Computational Harmonic Analysis*, 42:88–116, 2017.
- 50 E. L. Lehmann and Joseph P. Romano. *Testing statistical hypotheses*. Springer Texts in Statistics. Springer, New York, 2005.
- 51 R. Levi, D. Ron, and R. Rubinfeld. Testing properties of collections of distributions. In *proceedings of the Second Symposium on Innovations in Computer Science*, ICS '11, pages 179–194, 2011.
- 52 Florian Mintert, Marek Kuś, and Andreas Buchleitner. Concurrence of mixed multipartite quantum states. *Phys. Rev. Lett.*, 95:260502, 2005.
- 53 A. Montanaro and R. de Wolf. A survey of quantum property testing. *Theory of Computing Graduate Surveys*, 7, 2016.
- 54 M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, 10th edition, 2011.
- 55 R. O'Donnell and J. Wright. Quantum spectrum testing. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing*, STOC '15, pages 529–538, 2015.
- 56 R. O'Donnell and J. Wright. Efficient quantum tomography. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing*, STOC '16, pages 899–912, 2016.
- 57 R. O'Donnell and J. Wright. Efficient quantum tomography ii. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing*, STOC '17, pages 962–974, 2017.
- 58 L. Paninski. A coincidence-based test for uniformity given very sparsely sampled discrete data. *IEEE Trans. Inf. Theor.*, 54(10):4750–4755, 2008.
- 59 R. Rubinfeld. Taming big probability distributions. *XRDS*, 19(1):24–28, 2012.
- 60 R. Rubinfeld and M. Sudan. Self-testing polynomial functions efficiently and over rational domains. In *Proceedings of the 3rd Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '92, pages 23–32, 1992.
- 61 R. Rubinfeld and M. Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM Journal on Computing*, 25(2):252–271, 1996.
- 62 I. N. Sanov. On the probability of large deviations of random variables. *Mat. Sbornik*, 42:11–44, 1957.
- 63 G. Valiant and P. Valiant. Estimating the unseen: An  $n/\log(n)$ -sample estimator for entropy and support size, shown optimal via new clts. In *Proceedings of the Forty-third Annual ACM Symposium on Theory of Computing*, STOC '11, pages 685–694, 2011.
- 64 G. Valiant and P. Valiant. The power of linear estimators. In *Proceedings of the 2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, FOCS '11, pages 403–412, 2011.
- 65 G. Valiant and P. Valiant. An automatic inequality prover and instance optimal identity testing. In *Proceedings of the 2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, FOCS '14, pages 51–60, 2014.
- 66 G. Valiant and P. Valiant. Instance optimal learning of discrete distributions. In *Proceedings of the Forty-eighth Annual ACM Symposium on Theory of Computing*, STOC '16, pages 142–155, 2016.
- 67 G. Valiant and P. Valiant. Estimating the unseen: Improved estimators for entropy and other properties. *Journal of the ACM*, 64(6), 2017.
- 68 P. Valiant. Testing symmetric properties of distributions. In *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*, STOC '08, pages 383–392, 2008.
- 69 V. Voroninski. Quantum tomography from few full-rank observables, 2013. [arXiv:1309.7669](#).
- 70 Y. Wu and P. Yang. Minimax rates of entropy estimation on large alphabets via best polynomial approximation. *IEEE Transactions on Information Theory*, 62(6):3702–3720, 2016.
- 71 Nengkun Yu. Quantum closeness testing: A streaming algorithm and applications, 2020. [arXiv:1904.03218](#).



# Understanding the Relative Strength of QBF CDCL Solvers and QBF Resolution

Olaf Beyersdorff 

Friedrich Schiller Universität Jena, Germany  
olaf.beyersdorff@uni-jena.de

Benjamin Böhm 

Friedrich Schiller Universität Jena, Germany  
benjamin.boehm@uni-jena.de

---

## Abstract

QBF solvers implementing the QCDCL paradigm are powerful algorithms that successfully tackle many computationally complex applications. However, our theoretical understanding of the strength and limitations of these QCDCL solvers is very limited.

In this paper we suggest to formally model QCDCL solvers as proof systems. We define different policies that can be used for decision heuristics and unit propagation and give rise to a number of sound and complete QBF proof systems (and hence new QCDCL algorithms). With respect to the standard policies used in practical QCDCL solving, we show that the corresponding QCDCL proof system is incomparable (via exponential separations) to Q-resolution, the classical QBF resolution system used in the literature. This is in stark contrast to the propositional setting where CDCL and resolution are known to be p-equivalent.

This raises the question what formulas are hard for standard QCDCL, since Q-resolution lower bounds do not necessarily apply to QCDCL as we show here. In answer to this question we prove several lower bounds for QCDCL, including exponential lower bounds for a large class of random QBFs.

We also introduce a strengthening of the decision heuristic used in classical QCDCL, which does not necessarily decide variables in order of the prefix, but still allows to learn asserting clauses. We show that with this decision policy, QCDCL can be exponentially faster on some formulas.

We further exhibit a QCDCL proof system that is p-equivalent to Q-resolution. In comparison to classical QCDCL, this new QCDCL version adapts both decision and unit propagation policies.

**2012 ACM Subject Classification** Theory of computation → Proof complexity

**Keywords and phrases** CDCL, QBF, QCDCL, proof complexity, resolution, Q-resolution

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.12

**Related Version** This is an extended abstract. A full version of the paper is available at <https://eccc.weizmann.ac.il/report/2020/053/>.

**Funding** *Olaf Beyersdorff*: John Templeton Foundation (grant no. 60842), Carl Zeiss Foundation.

## 1 Introduction

SAT solving has revolutionised the way we perceive and approach computationally complex problems. While traditionally, NP-hard problems were considered computationally intractable, today SAT solvers routinely and successfully solve instances of NP-hard problems from virtually all application domains, and in particular problem instances of industrial relevance [53]. Starting with the classic DPLL algorithm from the 1960s [25, 26], there have been a number of milestones in the evolution of SAT solving, but clearly one of the breakthrough achievements was the introduction of clause learning in the late 1990s, leading to the paradigm of *conflict-driven clause learning* (CDCL) [43, 55], the predominant technique



© Olaf Beyersdorff and Benjamin Böhm;  
licensed under Creative Commons License CC-BY  
12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 12; pp. 12:1–12:20



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



of modern SAT solving. CDCL ingeniously combines a number of crucial ingredients, among them variable decision heuristics, unit propagation, clause learning from conflicts, and restarts (cf. [42] for an overview).

Inspired by the success of SAT solving, many researchers have concentrated on the task to extend the reach of these technologies to computationally even more challenging settings with *quantified Boolean formulas* (QBF) receiving key attention. As a PSPACE-complete problem, the satisfiability problem for QBFs encompasses all problems from the polynomial hierarchy and allows to encode many problems far more succinctly than in propositional logic (cf. [51] for applications).

One of the main techniques in QBF solving is the propositional CDCL technique, lifted to QBF in the form of QCDCL [56]. However, solving QBFs presents additional challenges as the quantifier type of variables (existential and universal) needs to be taken into account as well as the variable dependencies stemming from the quantifier prefix.<sup>1</sup> This particularly impacts the variable selection heuristics and details of the unit propagation within QCDCL. In addition to QCDCL there are further QBF solving techniques, exploiting QBF features absent in SAT, such as expanding universal variables in expansion solving [36] and dependency schemes in dependency-aware solving [40, 47, 52]. Compared to SAT solving, QBF solving is still at an earlier stage. However, QBF solving has seen huge improvements during the past 15 years [49], and there are problems of practical relevance where QBF solvers outperform SAT solvers [28].

The enormous success of SAT and QBF solving of course raises theoretical questions of utmost importance: why are these solvers so successful and what are their limitations? The main approach through understanding these questions comes from proof complexity [20, 46]. The central problem in proof complexity is to determine the size of the smallest proof for a given formula in a specified proof system, typically defined through a set of axioms and inference rules. Traces of runs of SAT/QBF solvers on unsatisfiable instances yield proofs of unsatisfiability, whereby each solver implicitly defines a proof system. In particular, SAT solvers implementing the DPLL and CDCL paradigms are based on resolution [46], which is arguably the most studied proof system in proof complexity.

*Propositional resolution* operates on clauses and uses the resolution rule

$$\frac{C \vee x \quad D \vee \bar{x}}{C \vee D} \quad (1)$$

as its only inference rule to derive a new clause  $C \vee D$  from the two parent clauses  $C \vee x$  and  $D \vee \bar{x}$ .<sup>2</sup> There is a host of lower bounds and lower bound techniques available for propositional resolution (cf. [5, 38, 50] for surveys).

While it is relatively easy to see that the classic DPLL branching algorithm [25, 26] exactly corresponds to tree-like resolution (where resolution derivations are in form of a tree), the *relation between CDCL and resolution* is far more complex. On the one hand, resolution proofs can be generated efficiently from traces of CDCL runs on unsatisfiable formulas [4], a crucial observation being that learned clauses are derivable by resolution [4, 43]. The opposite simulation is considerably more difficult, with a series of works [1, 4, 33, 48] culminating in the result that CDCL can efficiently simulate arbitrary resolution proofs, i.e., resolution and CDCL are equivalent. This directly implies that all known lower bounds for proof size in resolution translate into lower bounds for CDCL running time. In addition, other measures such as proof space model memory requirements of SAT solvers, thereby implying lower bounds on memory consumption, in particular when considering time-space tradeoffs [45].

<sup>1</sup> In this paper we focus on prenex QBFs with a CNF matrix.

<sup>2</sup> We denote such a resolution inference with pivot  $x$  by  $(C \vee x) \stackrel{x}{\bowtie} (D \vee \bar{x})$  throughout the paper.

Exciting as this equivalence between CDCL and resolution is from a theoretical point of view, it has to be interpreted with care. Proof systems are inherently non-deterministic procedures, while CDCL algorithms are largely deterministic (some randomisation might occasionally be used). To overcome this discrepancy, the simulations of resolution by CDCL [4, 48] use arbitrary decision heuristics and perform excessive restarts, both of which diverge from practical CDCL policies. Indeed, in very recent work [54] it was shown that CDCL with practical decision heuristics such as VSIDS [55] is exponentially weaker than resolution, and similar results have been obtained for further decision heuristics [44]. Regarding restarts there is intense research aiming to determine the power of CDCL without restarts from a proof complexity perspective (cf. [19, 21]).

On the QBF level, this naturally raises the question *what proof system corresponds to QCDCL*. As in propositional proof complexity, QBF resolution systems take a prominent place in the QBF proof system landscape, with the basic and historically first **Q-resolution** system [37] receiving key attention. **Q-resolution** is a refutational system that proves the falsity of fully quantified prenex QBFs with a CNF matrix (QCNFs). The system allows to use the propositional resolution rule (1) under the conditions that the pivot  $x$  is an existential variable and the resolvent  $C \vee D$  is non-tautological. In addition, **Q-resolution** uses a *universal reduction rule*

$$\frac{C \vee u}{C}, \quad (2)$$

where  $u$  is a universal literal that in the quantifier prefix is quantified right of all variables in  $C$ , i.e., none of the literals in  $C$  depends on  $u$ . For **Q-resolution** we have a number of lower bounds [3, 8, 12] as well as lower bound techniques, some of them lifted from propositional proof complexity [13, 15], but more interestingly some of them genuine to the QBF domain [8, 10] that unveil deep connections between proof size and circuit complexity [11, 16], unparalleled in the propositional domain.

Unlike in the relation between SAT and CDCL, it has been open whether QCDCL runs can be efficiently translated into **Q-resolution**. Instead, QCDCL runs can be simulated by the stronger QBF resolution system of **long-distance Q-resolution** [2, 56]. In fact, this system originates from solving, where it was noted that clauses learned from QCDCL conflicts can be derived in **long-distance Q-resolution** [56]. **Long-distance Q-resolution** implements a more liberal use of the resolution rule (1), which allows to derive certain tautologies. In general, allowing to derive tautologies with (1) is unsound. However, the tautologies allowed in **long-distance Q-resolution** do not present problems for soundness and are exactly those clauses needed when learning clauses in QCDCL. Hence **long-distance Q-resolution** simulates QCDCL [2, 56]. However, it is known that **long-distance Q-resolution** allows exponentially shorter proofs than **Q-resolution** for some QBFs [8, 9, 27].

We also remark that there are further QBF resolution systems (cf. [18] for an overview) and even stronger QBF calculi [11, 14, 23, 34]. Some of these correspond to other solving approaches in QBF, such as the system  $\forall\text{Exp}+\text{Res}$  [36] that captures expansion QBF solving [6].

In summary, it is fair to say that the relations between QCDCL solving and QBF resolution (either **Q-resolution** or **long-distance Q-resolution**) are *currently not well understood*. In particular, an analogue of the equivalence of CDCL SAT solving and propositional resolution [1, 4, 48] is currently absent in the QBF domain. This brings us to the topic of this paper. However, rather than giving an overview of our results in this introduction, we will describe our results in Sections 3 to 7, after stating some preliminaries in Section 2. Most proofs will be omitted in this extended abstract due to space constraints.

## 2 Preliminaries

### 2.1 Propositional and quantified formulas

We will consider propositional and quantified formulas over a countable set of variables. Variables and negations of variables are called *literals*, i.e., for a variable  $x$  we can form two literals:  $x$  and its negation  $\bar{x}$ . Sometimes we write  $x^1$  instead of  $x$  and  $x^0$  instead of  $\bar{x}$ . We denote the corresponding variable as  $\text{var}(x) := \text{var}(\bar{x}) := x$ .

A *clause* is a disjunction  $\ell_1 \vee \dots \vee \ell_m$  of some literals  $\ell_1, \dots, \ell_m$ . We will sometimes view a clause as a set of literals, i.e., we will use the notation  $\ell \in C$  if the literal  $\ell$  is one of the literals in the clause  $C$ . If  $m = 1$ , we will often write  $(\ell_1)$  to emphasize the difference between literals and clauses. The *empty clause* is the clause consisting of zero literals, denoted by  $(\perp)$ . For reasons of consistency it is helpful to define an *empty literal*, denoted by  $\perp$  in our case. As a consequence, we have  $\perp \in (\perp)$ , although we define the empty clause as a clause with zero literals.

The negation of a clause  $C = \ell_1 \vee \dots \vee \ell_m$  is called a *term*, i.e., terms are conjunctions  $\bar{\ell}_1 \wedge \dots \wedge \bar{\ell}_m$  of literals. Similarly terms can be considered as sets of literals. A *CNF* (*conjunctive normal form*) is a conjunction of clauses.

Let  $C = \ell_1 \vee \dots \vee \ell_m$ . We define  $\text{var}(C) := \{\text{var}(\ell_1), \dots, \text{var}(\ell_m)\}$ . For a CNF  $\phi = C_1 \wedge \dots \wedge C_n$  we define  $\text{var}(\phi) := \bigcup_{i=1}^n \text{var}(C_i)$ .

A clause or a set  $C$  of literals is called *tautological*, if there is a variable  $x$  with  $x, \bar{x} \in C$ .

An *assignment*  $\sigma$  of a set of variables  $X$  is a non-tautological set of literals, such that for all  $x \in X$  there is  $\ell \in \sigma$  with  $\text{var}(\ell) = x$ . The restriction of a clause  $C$  by an assignment  $\sigma$  is defined as

$$C|_\sigma := \begin{cases} \top \text{ (true)} & \text{if } C \cap \sigma \neq \emptyset, \\ \bigvee_{\substack{\ell \in C \\ \ell \notin \sigma}} \ell & \text{otherwise.} \end{cases}$$

For example, let  $C = t \vee x \vee y \vee \bar{z}$  and define the assignment  $\sigma := \{\bar{x}, z, w\}$ . Then we have  $C|_\sigma = t \vee y$ . Note that the set of assigned variables might differ from  $\text{var}(C)$ . In our case,  $\sigma$  is an assignment of the set  $X := \{x, z, w\}$ .

One can interpret  $\sigma$  as an operator that sets all literals from  $\sigma$  to the Boolean constant 1. We denote the set of assignments of  $X$  by  $\langle X \rangle$ . A CNF  $\phi$  *entails* another CNF  $\psi$  if each assignment that satisfies  $\phi$  also satisfies  $\psi$  (denoted by  $\phi \models \psi$ ).

A *QBF* (*quantified Boolean formula*)  $\Phi = Q \cdot \phi$  is a propositional formula  $\phi$  (also called *matrix*) together with a *prefix*  $Q$ . A prefix  $Q_1 x_1 Q_2 x_2 \dots Q_k x_k$  consists of variables  $x_1, \dots, x_k$  and quantifiers  $Q_1, \dots, Q_k \in \{\exists, \forall\}$ . We obtain an equivalent formula if we unite adjacent quantifiers of the same type. Therefore we can always assume the prefix to be in the form

$$Q = Q'_1 X_1 Q'_2 X_2 \dots Q'_s X_s$$

with nonempty sets of variables  $X_1, \dots, X_s$  and quantifiers  $Q'_1, \dots, Q'_s \in \{\exists, \forall\}$  such that  $Q'_i \neq Q'_{i+1}$  for  $i \in [s-1]$ . For a variable  $x$  in  $Q$  we denote the *quantifier level* with respect to  $Q$  by  $\text{lv}(x) = \text{lv}_\Phi(x) = i$ , if  $x \in X_i$ . Variables from  $\Phi$  are called *existential*, if the corresponding quantifier is  $\exists$ , and *universal* if the quantifier is  $\forall$ . We denote the set of existential variables from  $\Phi$  by  $\text{var}_\exists(\Phi)$ , and the set of universal variables by  $\text{var}_\forall(\Phi)$ .

A QBF with CNF matrix is called a *QCNF*. We require that all clauses from a matrix of a QCNF are non-tautological, otherwise we just delete these clauses. This requirement is crucial for the correctness of the derivation rules we define later for QBF proof systems. Since we will only discuss refutational proof systems, we will always assume that all QCNFs we consider are false.

A QBF can be interpreted as a game between two players: The  $\exists$ -player and the  $\forall$ -player. These players have to assign the respective variables one by one along the quantifier order from left to right. The  $\forall$ -player wins the game if and only if the matrix of the QBF gets falsified by this assignment. It is well known that for every false QBF  $\Phi = \mathcal{Q} \cdot \phi$  there exists a winning strategy for the  $\forall$ -player.

## 2.2 Q-resolution and long-distance Q-resolution

Let  $C_1$  and  $C_2$  be two clauses of a QCNF  $\Phi$  and let  $\ell$  be an existential literal with  $\text{var}(\ell) \notin \text{var}(C_1) \cup \text{var}(C_2)$ . The *resolvent* of  $C_1 \vee \ell$  and  $C_2 \vee \bar{\ell}$  over  $\ell$  is defined as

$$(C_1 \vee \ell) \overset{\ell}{\bowtie} (C_2 \vee \bar{\ell}) := C_1 \vee C_2.$$

Let  $C := u_1 \vee \dots \vee u_m \vee x_1 \vee \dots \vee x_n \vee v_1 \vee \dots \vee v_s$  be a clause from  $\Phi$ , where  $u_1, \dots, u_m, v_1, \dots, v_s$  are universal literals,  $x_1, \dots, x_n$  are existential literals and

$$\{v \in C : v \text{ is universal and } \text{lv}(v) > \text{lv}(x_i) \text{ for all } i \in [n]\} = \{v_1, \dots, v_s\}.$$

Then we can perform a *reduction* step and obtain

$$\text{red}(C) := u_1 \vee \dots \vee u_m \vee x_1 \vee \dots \vee x_n.$$

For a CNF  $\phi = \{C_1, \dots, C_k\}$  we define  $\text{red}(\phi) := \{\text{red}(C_1), \dots, \text{red}(C_k)\}$ .

Q-resolution [37] is a refutational proof system for false QCNFs. A Q-resolution proof  $\pi$  of a clause  $C$  from a QCNF  $\Phi = \mathcal{Q} \cdot \phi$  is a sequence of clauses  $\pi = C_1, \dots, C_m$  with  $C_m = C$ . Each  $C_i$  has to be derived by one of the following three rules:

- *Axiom*:  $C_i \in \phi$ ;
- *Resolution*:  $C_i = C_j \overset{x}{\bowtie} C_k$  for some  $j, k < i$  and  $x \in \text{var}_{\exists}(\Phi)$ , and  $C_i$  is non-tautological;
- *Reduction*:  $C_i = \text{red}(C_j)$  for some  $j < i$ .

Note that none of our axioms are tautological by definition. A *refutation* of a QCNF  $\Phi$  is a proof of the empty clause  $(\perp)$ .

For the simulating QCDCL runs, long-distance Q-resolution was introduced in [2, 56]. This extension of Q-resolution allows to derive universal tautologies under certain conditions. As in Q-resolution, there are three rules by which a clause  $C_i$  can be derived. The axiom and reduction rules are identical to Q-resolution, but the resolution rule is changed to

- *Resolution (long-distance)*:  $C_i = C_j \overset{x}{\bowtie} C_k$  for some  $j, k < i$  and  $x \in \text{var}_{\exists}(\Phi)$ . The resolvent  $C_i$  is allowed to contain a tautology  $u \vee \bar{u}$  if  $u$  is a universal variable. If  $u \in \text{var}(C_j) \cap \text{var}(C_k)$ , then we additionally require  $\text{lv}(u) > \text{lv}(x)$ .

Note that a long-distance Q-resolution proof without tautologies is just a Q-resolution proof.

Creating universal tautologies without any assumptions is unsound in general. For example, consider the true QCNF  $\Psi := \forall u \exists x \cdot (u \vee \bar{x}) \wedge (\bar{u} \vee x)$ . There is a winning strategy for the  $\exists$ -player by assigning  $x$  equal to  $u$ . Hence, the step  $\text{red}((u \vee \bar{x}) \overset{x}{\bowtie} (\bar{u} \vee x)) = (\perp)$  is unsound since we resolved over an existential literal  $x$  with  $\text{lv}_{\Psi}(x) > \text{lv}_{\Psi}(u)$  while generating  $u \vee \bar{u}$ .

### 3 Our framework: versions of QCDCL as formal proof systems

We now start to describe the framework for our results. Technically, this paper hinges on the formalisation of QCDCL solving as precisely defined proof systems, which can subsequently be analysed from a proof-complexity perspective. For this we need to formally define central ingredients of QCDCL solving, including trails, decision policies, unit propagation, and clause learning (cf. [18] for background). For decisions and unit propagation we will consider different policies: those corresponding to QCDCL solving in practice and new policies, yet unexplored. We will show that the corresponding QCDCL proof systems are all sound and complete.

We start with defining trails, decisions, unit propagations and our collection of policies.

► **Definition 1** (trails and policies for decision/unit propagation). *Let  $\Phi = \mathcal{Q} \cdot \phi$  be a QCNF in  $n$  variables. A trail  $\mathcal{T}$  for  $\Phi$  is a sequence of literals (or  $\perp$ ) of variables from  $\Phi$  with specific properties. We distinguish two types of literals in  $\mathcal{T}$ : decision literals, that can be both existential and universal, and propagated literals, that are either existential or  $\perp$ . Most of the time we write a trail  $\mathcal{T}$  as*

$$\mathcal{T} = (p_{(0,1)}, \dots, p_{(0,g_0)}; \mathbf{d}_1, p_{(1,1)}, \dots, p_{(1,g_1)}; \dots; \mathbf{d}_r, p_{(r,1)}, \dots, p_{(r,g_r)}).$$

We typically denote decision literals by  $d_i$  and propagated literals by  $p_{(i,j)}$ . To emphasize decisions, we will set decision literals in the trail in **boldface** and put a semicolon at the end of each decision level. The literal  $p_{(i,j)}$  represents the  $j^{\text{th}}$  propagated literal at the  $i^{\text{th}}$  decision level, determined by the corresponding decision  $d_i$ . The decision level 0 is the only level without a decision literal. Similarly as with clauses, we can view  $\mathcal{T}$  as a set of literals or as an assignment and use the notation  $x \in \mathcal{T}$  if the literal  $x$  is contained in  $\mathcal{T}$ .

Let  $s \in \{0, \dots, r\}$  and  $t \in \{0, \dots, g_s\}$ . The subtrail of  $\mathcal{T}$  at time  $(s, t)$  is the trail consisting of all literals from the leftmost literal in  $\mathcal{T}$  up to (and including)  $p_{(s,t)}$ , if  $t \neq 0$ , or  $d_s$  otherwise. We denote this subtrail by  $\mathcal{T}[s, t]$ . The subtrail  $\mathcal{T}[0, 0]$  is the empty trail.

We impose some further requirements for  $\mathcal{T}$  to be a trail for a QCNF  $\Phi$ . The decisions have to be non-tautological and non-repeating, i.e., we require  $\text{var}(d_i) \neq \text{var}(d_k)$  for each  $i \neq k \in \{0, \dots, r\}$ . If  $\perp \in \mathcal{T}$ , then this must be the last (rightmost) literal in  $\mathcal{T}$ . In this case we say that  $\mathcal{T}$  has run into a conflict.

We define four policies, concerning the decision of literals, from which we can choose exactly one at a time:

- **LEV-ORD** - For each  $d_i \in \mathcal{T}$  we have  $lv(d_i) \leq lv(x)$  for all  $x \in \text{var}(\phi) \setminus \text{var}(\mathcal{T}[i-1, g_{i-1}])$ . This means that we have to decide the variables along the quantification order.
- **ASS-ORD** - We can decide a literal  $d_k$  if it is existential, or if it is universal and  $lv(d_1) \leq \dots \leq lv(d_k)$ .
- **ASS-R-ORD** - We can only decide an existential variable  $x$  next, if and only if we already decided all universal variables  $u$  with  $lv(u) < lv(x)$  before.
- **ANY-ORD** - We can choose any remaining literal as the next decision.

We define two more policies concerning unit propagation. Again, we have to choose exactly one:

- **RED** - For each  $p_{(i,j)} \in \mathcal{T}$  there has to be a clause  $C \in \phi$  such that  $\text{red}(C|_{\mathcal{T}[i,j-1]}) = (p_{(i,j)})$ .
- **NO-RED** - For each  $p_{(i,j)} \in \mathcal{T}$  there has to be a clause  $C \in \phi$  with  $C|_{\mathcal{T}[i,j-1]} = (p_{(i,j)})$ .

These clauses  $C$  as described in the unit-propagation policies are called antecedent clauses and will be denoted by  $\text{ante}_{\mathcal{T}}(p_{(i,j)}) := C$ . There could be more than one such suitable clause, in which case we will just choose one of them arbitrarily. The antecedent clauses clearly depend on the unit propagation policy we use.

The size of a trail  $\mathcal{T}$  is measured by  $|\mathcal{T}|$  (i.e., the cardinality of  $\mathcal{T}$  as a set). Because each trail can at most contain all variables, we have  $|\mathcal{T}| \in \mathcal{O}(n)$ .

We remark that QCDCL as used in practice employs the policies LEV-ORD and RED, and the decision policy ANY-ORD originates from CDCL.

The policies RED and NO-RED determine the notion of unit clauses, which are important for unit propagation.

► **Definition 2** (unit clauses). *Let  $C$  be a clause. In the policy RED, we call  $C$  a unit clause if  $\text{red}(C) = (x)$  for an existential literal  $x$  or  $x = \perp$ . Otherwise, for NO-RED, we call  $C$  a unit clause if  $C = (x)$  for an existential literal  $x$  or  $x = \perp$ .*

Note that  $(u)$  is not a unit clause under the policy NO-RED for a universal literal  $u$ .

In (Q)CDCL, whenever a trail  $\mathcal{T}$  runs into a conflict, i.e., a clause  $C$  from  $\Phi$  is falsified, we perform conflict analysis in the form of *clause learning*. This results in a clause  $D$  that follows from  $\Phi$  and describes a reason for the conflict. Such conflict clauses are obtained by performing resolution (for CDCL) and long-distance Q-resolution (for QCDCL), starting from the conflict clause  $C$  and resolving along the propagated variables in  $\mathcal{T}$  in reverse order (skipping resolution steps when the pivot is missing).

► **Definition 3** (learnable clauses). *Let  $\Phi = \mathcal{Q} \cdot \phi$  be a QCNF and let*

$$\mathcal{T} = (p_{(0,1)}, \dots, p_{(0,g_0)}; \mathbf{d}_1, p_{(1,1)}, \dots, p_{(1,g_1)}; \dots; \mathbf{d}_r, p_{(r,1)}, \dots, p_{(r,g_r)})$$

*be a trail with  $p_{(r,g_r)} = \perp$  that follows policies  $P \in \{\text{LEV-ORD}, \text{ASS-ORD}, \text{ASS-R-ORD}, \text{ANY-ORD}\}$  and  $R \in \{\text{RED}, \text{NO-RED}\}$ . We call a clause learnable from  $\mathcal{T}$  if it appears in the sequence*

$$\mathcal{L}_{\mathcal{T}} := (C_{(r,g_r)}, \dots, C_{(r,1)}, \dots, C_{(1,g_1)}, \dots, C_{(1,1)}, C_{(0,g_0)}, \dots, C_{(0,1)})$$

*where  $C_{(r,g_r)} := \text{red}(\text{ante}(p_{(r,g_r)}))$ ,*

$$C_{(i,j)} := \begin{cases} \text{red}\left(C_{(i,j+1)} \stackrel{p_{(i,j)}}{\boxtimes} \text{red}(\text{ante}(p_{(i,j)}))\right) & \text{if } \bar{p}_{(i,j)} \in C_{(i,j+1)}, \\ C_{(i,j+1)} & \text{otherwise} \end{cases}$$

*for  $i \in \{0, \dots, r\}$ ,  $j \in [g_i - 1]$ , and*

$$C_{(i,g_i)} := \begin{cases} \text{red}\left(C_{(i+1,1)} \stackrel{p_{(i,g_i)}}{\boxtimes} \text{red}(\text{ante}(p_{(i,g_i)}))\right) & \text{if } \bar{p}_{(i,g_i)} \in C_{(i+1,1)}, \\ C_{(i+1,1)} & \text{otherwise} \end{cases}$$

*for  $i \in \{0, \dots, r-1\}$ .*

Note that clause learning works independently from the used policy. Even if we choose the policy NO-RED, we might have to make reduction steps in this process. After the construction of each trail  $\mathcal{T}$  we will choose to learn exactly one clause from  $\mathcal{L}_{\mathcal{T}}$ . The actual choice represents a kind of nondeterminism in the learning process.

Next we formalise natural trails, where we are not allowed to skip unit propagations.



► **Definition 4** (natural trails). *We call a trail  $\mathcal{T}$  natural, if the following holds: For any time  $(s, t)$ ,  $s \in \{0, \dots, r\}$  and  $t \in [g_s]$ , if  $\{D_1, \dots, D_h\}$  are all clauses from the corresponding QCNF that become unit clauses  $(\ell_1), \dots, (\ell_h)$  under the assignment  $\mathcal{T}[s, t-1]$ , then the next propagated literal has to be one of the  $\ell_i$  together with  $D_i$  as antecedent clause. If one of the  $\ell_i$  is  $\perp$ , then we have to choose this  $\ell_i$ . I.e., conflicts have higher priority.*

The next definition presents the main framework for this paper. Having defined trails in a general sense, we specify how a trail can be generated during a QCDCL run. We introduce the notion of QCDCL-based proofs consisting of three components: the naturally created trails, the clauses learned from each trail, and the proof of each learned clause.

► **Definition 5** (QCDCL proof systems). *Let  $\Phi = \mathcal{Q} \cdot \phi$  be a QCNF in  $n$  variables. We call a triple of sequences*

$$\iota = (\underbrace{(\mathcal{T}_1, \dots, \mathcal{T}_m)}_{=: \theta(\iota)}, \underbrace{(C_1, \dots, C_m)}_{=: \lambda(\iota)}, \underbrace{(\pi_1, \dots, \pi_m)}_{=: \rho(\iota)})$$

a QCDCL $_R^P$  proof from  $\Phi$  of a clause  $C$  for  $P \in \{\text{LEV-ORD}, \text{ASS-ORD}, \text{ASS-R-ORD}, \text{ANY-ORD}\}$  and  $R \in \{\text{RED}, \text{NO-RED}\}$ , if for all  $i \in [m]$  the trail  $\mathcal{T}_i$  follows the policies  $P$  and  $R$  and uses the QCNF  $\mathcal{Q} \cdot (\phi \cup \{C_1, \dots, C_{i-1}\})$ , where  $C_j \in \mathcal{L}_{\mathcal{T}_j}$  is a clause learnable from  $\mathcal{T}_j$  and  $C_m = C$ . Each  $\pi_i$  is the derivation of the clause  $C_i$  from  $\mathcal{Q} \cdot (\phi \cup \{C_1, \dots, C_{i-1}\})$  as defined recursively in Definition 3. We will denote  $(\mathcal{T}_1, \dots, \mathcal{T}_m)$  by  $\theta(\iota)$ ,  $(C_1, \dots, C_m)$  by  $\lambda(\iota)$  and  $(\pi_1, \dots, \pi_m)$  by  $\rho(\iota)$ . Note that all these trails need to run into a conflict in order to start clause learning. If  $C = (\perp)$  we call  $\iota$  a refutation.

We also require that  $\mathcal{T}_1$  is natural and for each  $i \in \{2, \dots, m\}$  there exist indices  $(s, t)$  such that the following holds:

- $\mathcal{T}_i[s, t] = \mathcal{T}_{i-1}[s, t]$ .
- For each subtrail  $\mathcal{T}_i[a, b]$  with  $\mathcal{T}_i[s, t] \subseteq \mathcal{T}_i[a, b]$  and  $\perp \notin \mathcal{T}_i[a, b]$  let  $D_1, \dots, D_h$  be all the clauses in  $\phi \cup \{C_1, \dots, C_{i-1}\}$  such that under the assignment  $\mathcal{T}_i[a, b]$  these clauses get unit (under the policy  $R$ ) with corresponding literals  $\ell_1, \dots, \ell_h$ . Then we have to propagate one of these literals next, i.e.,  $\ell_j \in \mathcal{T}_i[a, b+1]$  for some  $j \in [h]$ , and take the corresponding clause  $D_j$  as antecedent.
- In the situation above, if  $\perp \in \{\ell_1, \dots, \ell_h\}$ , then  $\perp \in \mathcal{T}_i[a, b+1]$ . I.e., we have to run into a conflict as soon as we find one.

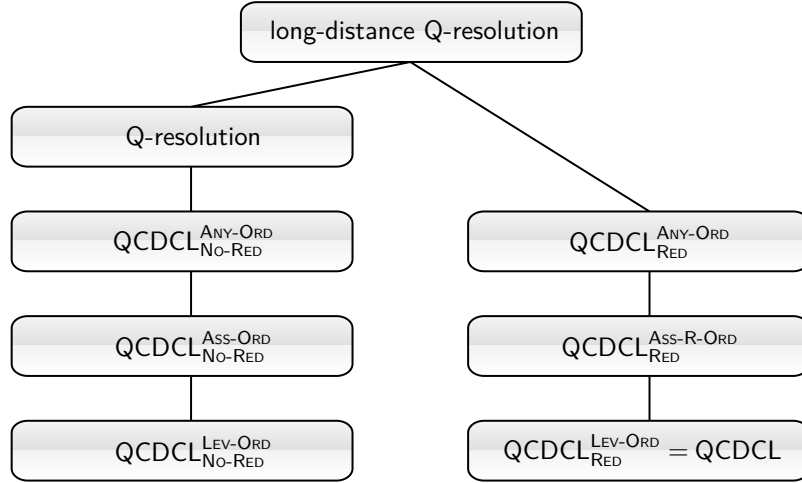
We call this process backtracking to  $\mathcal{T}_i[s, t]$ . Backtracking to  $\mathcal{T}_i[0, 0]$  is called restarting. The size of a proof  $\iota$  is measured by  $|\iota| := \sum_{j=1}^m |\mathcal{T}_j| \in \mathcal{O}(mn)$ .

The corresponding (refutational) proof system for false QCNFs is denoted QCDCL $_R^P$ . We will refer to these systems as QCDCL proof systems. A trail  $\mathcal{T}$  that follows the policies  $P$  and  $R$  is a QCDCL $_R^P$  trail.

Note that the first trail  $\mathcal{T}_1$  of each proof  $\iota$  is always natural.

Combining the two policies RED and NO-RED for unit propagation and the four policies ANY-ORD, LEV-ORD, ASS-ORD, and ASS-R-ORD, we obtain six QCDCL systems. These are depicted in Figure 1 (we are not interested in the systems QCDCL $_{\text{RED}}^{\text{ASS-ORD}}$  and QCDCL $_{\text{NO-RED}}^{\text{ASS-R-ORD}}$  since ASS-ORD and ASS-R-ORD would not be beneficial in these combinations). As mentioned, combining LEV-ORD with RED yields the standard QCDCL system, and we will also write QCDCL for QCDCL $_{\text{RED}}^{\text{LEV-ORD}}$ . The other five variants are introduced here for the first time.

The decision policies ASS-ORD and ASS-R-ORD might seem slightly unintuitive at first sight. We show that these policies guarantee the learning of so-called asserting clauses (Definition 6) in association with NO-RED resp. RED.



■ **Figure 1** Overview of the defined QCDCL proof systems. Lines denote p-simulations and follow by definition and Theorem 8.

A proof system  $P$  p-simulates a proof system  $S$  if each  $S$  proof can be efficiently transformed into a  $P$  proof of the same formula [24]. If the systems p-simulate each other, they are p-equivalent.

It will turn out that  $\pi_1, \dots, \pi_m$  in Definition 5 are in fact valid long-distance Q-resolution proofs. To prove this, we will argue that in proof systems with NO-RED we cannot derive any tautologies, while with RED we can at most derive universal tautologies.

Next we introduce *asserting learning schemes*. These are commonly used in practice since they guarantee a kind of progression in a run. These learning schemes are important to prevent a trail from backtracking too often.

► **Definition 6** (asserting clauses and asserting learning schemes). Let  $\Phi := \mathcal{Q} \cdot \phi$  be a QCNF in any of the defined QCDCL systems. Let

$$\mathcal{T} = (p_{(0,1)}, \dots, p_{(0,g_0)}; \mathbf{d}_1, p_{(1,1)}, \dots, p_{(1,g_1)}; \dots; \mathbf{d}_r, p_{(r,1)}, \dots, p_{(r,g_r)} = \perp)$$

be a trail which follows the corresponding policies and  $\mathcal{L}_{\mathcal{T}}$  the sequence of learnable clauses. A nonempty clause  $C \in \mathcal{L}_{\mathcal{T}}$  is called an asserting clause, if it becomes unit after backtracking, i.e., there exists a time  $(s, t)$  with  $s \in \{0, \dots, r-1\}$  and  $t \in [g_s]$  such that  $C|_{\mathcal{T}_{[s,t]}}$  is a unit clause under the corresponding system.

Let  $\mathbb{T}$  be the set of trails  $\mathcal{T}$  for  $\Phi$  such that  $\perp \in \mathcal{T}$ . A learning scheme  $\xi$  is a map with domain  $\mathbb{T}$ , which maps each  $\mathcal{T}$  to a clause  $\xi(\mathcal{T}) \in \mathcal{L}_{\mathcal{T}}$ .

A learning scheme  $\xi$  is called asserting if it maps to asserting clauses or  $(\perp)$  as long as  $\mathcal{L}_{\mathcal{T}}$  contains such.

It is not guaranteed that we will always find asserting clauses for trails. For example consider the false QCNF  $\forall u \exists x \cdot (u \vee x) \wedge (u \vee \bar{x}) \wedge (\bar{u} \vee x) \wedge (\bar{u} \vee \bar{x})$  and the trail  $\mathcal{T} = (\mathbf{x}; \mathbf{u}, \perp)$  under the system  $\text{QCDCL}_{\text{NO-RED}}^{\text{ANY-ORD}}$ . We can only learn the clause  $(\bar{u} \vee \bar{x})$ , which is non-unit under  $\mathcal{T}[0, 0] = \emptyset$ .

However, we can always learn asserting clauses when using one of the policies ASS-ORD or ASS-R-ORD, which is the reason why we introduced these policies.

► **Lemma 7.**

- Let  $\mathcal{T}$  be a trail under the policies ASS-ORD and NO-RED. If  $(\perp) \notin \mathcal{L}_{\mathcal{T}}$ , then there exists an asserting clause  $D \in \mathcal{L}_{\mathcal{T}}$ .
- Let  $\mathcal{T}$  be a trail under the policies ASS-R-ORD and RED. If  $(\perp) \notin \mathcal{L}_{\mathcal{T}}$ , then there exists an asserting clause  $D \in \mathcal{L}_{\mathcal{T}}$ .



We establish that all systems depicted in Figure 1 are sound and complete.

► **Theorem 8.** *All defined QCDCL proof systems are sound and complete QBF proof systems. In particular, all QCDCL calculi are  $p$ -simulated by long-distance Q-resolution and the proof systems with NO-RED are even  $p$ -simulated by Q-resolution.*

Soundness is shown via efficiently constructing long-distance Q-resolution proofs from QCDCL proofs. Crucially, when using the unit-propagation policy NO-RED, then no long-distance steps are actually needed and we just construct Q-resolution proofs. The resulting simulations are depicted in Figure 1. Simulations between the QCDCL calculi follow by definition. We remark already here that this simulation order simplifies further due to our results in the following sections (cf. Figure 2).

Proving that QCDCL decisions do not necessarily need to follow the order of quantification (as is done in practical QCDCL with policy LEV-ORD), might be a somewhat surprising discovery. It seems to us that inside the QBF community there is the wide-spread belief that following the quantification order in decisions is needed for soundness (cf. e.g. [30, 41, 56]).<sup>3</sup> While this is true for QDPLL [22, 30],<sup>4</sup> it is actually not needed in QCDCL: the quantification order is immaterial for the decisions as long as the quantification order is correctly taken into account when deriving learned clauses (Theorem 8).<sup>5</sup> Hence our theoretical work also opens the door towards *new solving approaches in practice* (cf. the discussion in Section 8).

From a theoretical point of view, formalising the QCDCL ingredients into proof systems enables a precise proof-theoretic analysis of the QCDCL systems and their comparison to Q-resolution. This will be the underlying feature of our results in the following two sections, showing the incomparability of Q-resolution and QCDCL (Section 4) and the lower bounds for QCDCL (Section 5). We will use it further to obtain a version of QCDCL that is even  $p$ -equivalent to Q-resolution (Section 6).

## 4 QCDCL and Q-resolution are incomparable

This section establishes that QCDCL and Q-resolution are incomparable by exponential separations, i.e., there exist QBFs that are easy for QCDCL, but require exponential-size Q-resolution refutations, and vice versa. As explained above, this is in stark contrast to the propositional setting, where CDCL and resolution are equivalent.

► **Theorem 9.** *The systems Q-resolution and QCDCL are incomparable.*

Proving Theorem 9 requires two families of QBFs. For the first we take the parity formulas.

► **Definition 10** ([12]). *The QCNF QParity<sub>n</sub> consists of the prefix  $\exists x_1 \dots x_n \forall z \exists t_2 \dots t_n$  and the matrix*

$$\begin{aligned} & x_1 \vee x_2 \vee \bar{t}_2, \quad x_1 \vee \bar{x}_2 \vee t_2, \quad \bar{x}_1 \vee x_2 \vee t_2, \quad \bar{x}_1 \vee \bar{x}_2 \vee \bar{t}_2, \\ & x_i \vee t_{i-1} \vee \bar{t}_i, \quad x_i \vee \bar{t}_{i-1} \vee t_i, \quad \bar{x}_i \vee t_{i-1} \vee t_i, \quad \bar{x}_i \vee \bar{t}_{i-1} \vee \bar{t}_i, \\ & t_n \vee z, \quad \bar{t}_n \vee \bar{z} \end{aligned}$$

for  $i \in \{2, \dots, n\}$ .

<sup>3</sup> In fact we thought so too, prior to this paper.

<sup>4</sup> The fact that the earlier QDPLL algorithm [22] needs to obey the quantifier order might have been the reason why this policy was adopted in QCDCL as well [56].

<sup>5</sup> We note, however, that the approach of *dependency learning* [47] starts with an empty set of dependency conditions (cf. [7, 52] for background on dependencies) and incrementally learns new dependencies. As decisions only need to respect the learned dependencies, they can initially be made out of order [47].

The formulas assert that there is an input  $x_1, \dots, x_n$  such that the parity  $\bigoplus_{i \in [n]} x_i$  is not equal to  $z$ . Since  $z$  is universally quantified, this means that  $\bigoplus_{i \in [n]} x_i$  should be neither 0 nor 1, an obvious contradiction. The parity computation is encoded by using variables  $t_i$  for the prefix sums  $\bigoplus_{j \in [i]} x_j$ . Using strategy extraction for Q-resolution [2, 12] and the result that the parity function is hard for bounded-depth circuits [29, 32], one can show that the  $\text{QParity}_n$  formulas require exponential-size Q-resolution refutations [12].

Here we show that  $\text{QParity}_n$  is easy for QCDCL.

► **Proposition 11.**  *$\text{QParity}_n$  has polynomial-size proofs in QCDCL.*

This requires to construct specific trails and clauses learned from these trails that together comprise a short QCDCL proof of the formulas.

For the opposite separation we consider the following QBFs:

► **Definition 12.** *Let  $\text{PHP}_n^{n+1}$  be the set of clauses for the pigeonhole principle with  $n$  holes and  $n + 1$  pigeons using variables  $x_1, \dots, x_{s_n}$ . Let  $\text{Trapdoor}_n$  be the QCNF with the prefix  $\exists y_1, \dots, y_{s_n} \forall w \exists t, x_1, \dots, x_{s_n} \forall u$  and the matrix*

$$\text{PHP}_n^{n+1}(x_1, \dots, x_{s_n}) \tag{3}$$

$$\bar{y}_i \vee x_i \vee u, \quad y_i \vee \bar{x}_i \vee u \tag{4}$$

$$y_i \vee w \vee t, \quad y_i \vee w \vee \bar{t}, \quad \bar{y}_i \vee w \vee t, \quad \bar{y}_i \vee w \vee \bar{t} \tag{5}$$

for  $i = 1, \dots, s_n$ .

We show that these formulas  $\text{Trapdoor}_n$  require exponential-size QCDCL refutations. In QCDCL, variables have to be decided in order of the quantifier prefix, hence each QCDCL trail for  $\text{Trapdoor}_n$  has to start with the  $y$  variables, which by unit propagation (used together with universal reduction) propagates  $x_i = y_i$  for  $i \in [s_n]$  by clauses (4). Therefore the trail runs into a conflict on the PHP clauses (3). This happens repeatedly, forcing QCDCL to produce a resolution refutation of the clauses (3), which by the propositional resolution lower bound by Haken [31] has to be of exponential size.

► **Proposition 13.** *The QCNFs  $\text{Trapdoor}_n$  require exponential-size  $\text{QCDCL}_{\text{RED}}^{\text{LEV-ORD}}$  refutations.*

On the other hand, it is easy to obtain short Q-resolution refutations of  $\text{Trapdoor}_n$  by just using the clauses (5).

► **Proposition 14.** *The QCNFs  $\text{Trapdoor}_n$  have constant-size Q-resolution refutations.*

This establishes the separation of QCDCL and Q-resolution. We remark that in earlier work, Janota [35] showed that QCDCL with a specific asserting learning scheme requires large running time on some class of QBFs, whereas the same formulas are easy for Q-resolution. Of course, this raises the question whether another learning scheme might produce short QCDCL runs. In contrast, our Theorem 9 rules out any simulation of Q-resolution by QCDCL (or vice versa), regardless of the learning scheme used.

## 5 Lower bounds for QCDCL

The incomparability of Q-resolution and QCDCL raises the immediate question of what formulas are hard for QCDCL. Previous research has largely concentrated on showing lower bounds for Q-resolution (e.g. [8, 12, 37]). However, by our results from the last section, these lower bounds do not necessarily apply to QCDCL, and prior to this paper no dedicated lower bounds for QCDCL (with arbitrary learning schemes) were known.

Here we show that several formulas from the QBF literature, including the equality formulas and a large class of random QBFs [8] are indeed hard for QCDCL.

We start by defining a proof system in which we can analyse hardness in classical QCDCL.

► **Definition 15.** *We call a long-distance Q-resolution proof  $\pi$  of a clause  $C$  from a QCNF  $\Phi$  a long-distance QCDCL resolution proof of  $C$  from  $\Phi$ , if there exists a  $\text{QCDCL}_{\text{RED}}^{\text{LEV-ORD}}$  proof  $\iota$  of  $C$  from  $\Phi$  such that the long-distance Q-resolution proof  $\pi$  is obtained by pasting together the sub-proofs  $(\pi_1, \dots, \pi_m)$  from  $\iota$  (cf. Definition 5).*

The system long-distance QCDCL resolution identifies a fragment of long-distance Q-resolution, which collects all long-distance Q-resolution proofs that appear in  $\text{QCDCL}_{\text{RED}}^{\text{LEV-ORD}}$  derivations. By definition therefore, long-distance QCDCL resolution and  $\text{QCDCL}_{\text{RED}}^{\text{LEV-ORD}}$  are p-equivalent proof systems.

Our next goal is to identify a whole class of QCNFs that witness the hardness of QCDCL.

The equality formulas from [8] are arguably one of the simplest families of QBFs that are interesting from a proof complexity perspective. The formula  $\text{Equality}_n$  is defined as the QCNF

$$\exists x_1 \dots x_n \forall u_1 \dots u_n \exists t_1 \dots t_n \cdot (\bar{t}_1 \vee \dots \vee \bar{t}_n) \wedge \bigwedge_{i=1}^n ((\bar{x}_i \vee \bar{u}_i \vee t_i) \wedge (x_i \vee u_i \vee t_i)).$$

These formulas are of the type  $\Sigma_3^b$ , i.e., they have two quantifier alternations starting with  $\exists$ .

Inspired by this construction, [8] considered a class of randomly generated QCNFs, again of type  $\Sigma_3^b$ .

► **Definition 16 ([8]).** *For each  $1 \leq i \leq n$  let  $C_i^{(1)}, \dots, C_i^{(cn)}$  be clauses picked uniformly at random from the set of clauses containing 1 literal from the set  $U_i = \{u_i^{(1)}, \dots, u_i^{(m)}\}$  and 2 literals from  $X_i = \{x_i^{(1)}, \dots, x_i^{(n)}\}$ . Define the randomly generated QCNF  $Q(n, m, c)$  as:*

$$Q(n, m, c) := \exists X_1, \dots, X_n \forall U_1, \dots, U_n \exists t_1, \dots, t_n \cdot \bigwedge_{i=1}^n \bigwedge_{j=1}^{cn} (\bar{t}_i \vee C_i^{(j)}) \wedge (t_1 \vee \dots \vee t_n).$$

Suitably choosing the parameters  $c$  and  $m$ , we obtain false QBFs with high probability.

Both the equality and the random formulas require exponential-size proofs in Q-resolution (the random formulas whp) [8]. This is shown in [8] via the size-cost-capacity technique, a semantically grounded QBF lower-bound technique that infers Q-resolution hardness for formulas  $\Phi_n$  (and in fact hardness for even stronger systems) from lower bounds for the size of countermodels for  $\Phi_n$ .

It is not clear how to directly apply this technique to QCDCL. Instead, we identify a property, which we term the *XT-property*, that we can use to lift hardness from Q-resolution to QCDCL.

► **Definition 17.** *Let  $\Phi$  be a QCNF of the form  $\exists X \forall U \exists T \cdot \phi$  with sets of variables  $X = \{x_1, \dots, x_a\}$ ,  $U = \{u_1, \dots, u_b\}$  and  $T = \{t_1, \dots, t_c\}$ .*

*We call a clause  $C$  in the variables of  $\Phi$*

- *T-clause, if  $\text{var}(C) \cap X = \emptyset$ ,  $\text{var}(C) \cap U = \emptyset$  and  $\text{var}(C) \cap T \neq \emptyset$ ,*
- *XT-clause, if  $\text{var}(C) \cap X \neq \emptyset$ ,  $\text{var}(C) \cap U = \emptyset$  and  $\text{var}(C) \cap T \neq \emptyset$ ,*
- *XUT-clause, if  $\text{var}(C) \cap X \neq \emptyset$ ,  $\text{var}(C) \cap U \neq \emptyset$  and  $\text{var}(C) \cap T \neq \emptyset$ .*

*We say that  $\Phi$  fulfils the XT-property if  $\phi$  contains no XT-clauses as well as no unit T-clauses and there do not exist two T-clauses  $C_1, C_2 \in \phi$  that are resolvable.*

Intuitively, this says that in a  $\Sigma_3^b$  formula  $\Phi$  with quantifier prefix of the form  $\exists X \forall U \exists T$  with blocks of variables  $X, U, T$ , there is no direct connection between the  $X$  and  $T$  variables, i.e.,  $\Phi$  does not contain clauses with  $X$  and  $T$  variables, but no  $U$  variables.

We can then prove that QCDCL runs on formulas with this  $XT$ -property can be efficiently transformed into Q-resolution refutations, not only into long-distance Q-resolution refutations.

We first show that under the  $XT$ -property we cannot derive any  $XT$ -clauses.

► **Lemma 18.** *It is not possible to derive  $XT$ -clauses by long-distance Q-resolution from a QCNF  $\Phi$  that fulfils the  $XT$ -property.*

**Proof.** Assume that we can derive an  $XT$ -clause  $C$  by a long-distance Q-resolution proof  $\pi$  from  $\Phi$ . Let  $D$  be the first  $XT$ -clause in  $\pi$  ( $D$  might be equal to  $C$ ). Since  $\Phi$  contains no  $XT$ -clauses as axioms, the last step before  $D$  has to be a resolution or reduction. A reduction is not possible since the reduced universal literal would have been blocked by a  $T$ -literal in  $D$ .

Therefore  $D$  is the resolvent of two preceding clauses  $D_1$  and  $D_2$ . If we resolve over an  $X$ -literal, then one of these clauses has to be an  $XT$ -clause. The same is true for a resolution over a  $T$ -literal. However, this contradicts the fact that  $D$  was the first  $XT$ -clause in  $\pi$ . ◀

The next lemma shows that under the  $XT$ -property it is also not possible to derive any non-axiomatic  $T$ -clauses.

► **Lemma 19.** *Let  $\Phi$  be a QCNF with the  $XT$ -property and let  $C$  be a  $T$ -clause derived by long-distance Q-resolution from  $\Phi$ . Then  $C$  is an axiom from  $\Phi$ .*

We will show later that we need to resolve two  $XUT$ -clauses over an  $X$ -literal in order to introduce tautologies. Now we prove that this is not possible in long-distance QCDCL resolution under the  $XT$ -property.

► **Lemma 20.** *It is not possible to resolve two  $XUT$ -clauses over an  $X$ -literal in a long-distance QCDCL resolution proof of a QCNF  $\Phi$  that fulfils the  $XT$ -property.*

**Proof.** Assume there is a long-distance QCDCL resolution proof  $\pi$  that contains such a resolution step over an  $X$ -literal  $x$ . Let  $C_1$  and  $C_2$  be the corresponding  $XUT$ -clauses. One of these clauses, say  $C_1$ , had to be an antecedent clause in a  $\text{QCDCL}_{\text{RED}}^{\text{LEV-ORD}}$  trail  $\mathcal{T}$  that implied  $x$ . Since our decisions in the trail are level-ordered and we did not skip any decisions, either  $x$  was propagated at decision level 0, or at a decision level in which we decided another  $X$ -literal.

Because  $C_1$  is an  $XUT$ -clause, we can find a  $T$ -literal  $t \in C_1$ . The literal  $\bar{t}$  must have been propagated before we implied  $x$  ( $\bar{t}$  could not have been decided because the decisions are level-ordered). That means that for the same trail we can find  $E := \text{ante}_{\mathcal{T}}(\bar{t})$ . Now,  $E$  cannot be a unit  $T$ -clause by the  $XT$ -property and Lemma 19. Hence  $E$  must contain further  $X$ -,  $U$ -, or  $T$ -literals. If  $E$  contains a  $U$ -literal, then we would have had to decide this  $U$ -literal before we use  $E$  as an antecedent clause, contradicting the level-order of our decisions. Also, this  $U$ -literal cannot be reduced since we want to imply a  $T$ -literal with the help of  $E$ . Therefore we conclude that  $E$  contains an  $X$ -literal or a  $T$ -literal. If  $E$  contains an  $X$ -literal, then  $E$  is an  $XT$ -clause, which is not possible by Lemma 18.

Therefore  $E$  contains at least another  $T$ -literal  $\ell \in E$ . As before, the literal  $\bar{\ell}$  was propagated before we implied  $\bar{t}$  and  $x$ . We set  $E' := \text{ante}_{\mathcal{T}}(\bar{\ell})$  and argue in the same way as with  $E$ . This process would repeat endlessly, which is a contradiction since we only have finitely many  $T$ -variables. ◀

Thus for formulas with the  $XT$ -property we can lift Q-resolution lower bounds to QCDCL, yielding the next theorem.

► **Theorem 21.** *If  $\Phi$  fulfils the  $XT$ -property and requires Q-resolution refutations of size  $s$ , then each QCDCL refutation of  $\Phi$  has size at least  $s$  as well.*

**Proof.** Let  $\pi$  be a long-distance QCDCL resolution refutation of  $\Phi$ . We show that  $\pi$  does not contain any tautological clause  $C$  and hence  $\pi$  is in fact a Q-resolution proof.

Assume that  $\pi$  contains some tautological clause  $C$ . W.l.o.g. let  $C$  be the first tautological clause in  $\pi$ . Clearly,  $C$  has to be derived by a resolution step over an  $X$ -literal. Let  $C_1$  and  $C_2$  be the parent clauses of  $C$ . Both of them contain some  $X$ -literals and some  $U$ -literals. They also have to contain  $T$ -literals, otherwise we would reduce all  $U$ -literals (in the learning process we reduce as soon as possible). Therefore  $C_1$  and  $C_2$  are both  $XUT$ -clauses that are resolved over an  $X$ -literal, which is not possible by Lemma 20.

Therefore such a clause  $C$  cannot exist. Hence each long-distance QCDCL resolution refutation of  $\Phi$  is even a Q-resolution refutation and the result follows. ◀

It is quite easy to check that both the equality formulas as well as the random formulas above have the  $XT$ -property. Thus we obtain:

► **Corollary 22.**

- Equality <sub>$n$</sub>  requires QCDCL refutations of size  $2^n$ .
- Let  $1 < c < 2$  be a constant and  $m \leq (1 - \epsilon) \log_2 n$  for some constant  $\epsilon > 0$ . With probability  $1 - o(1)$  the random QCNF  $Q(n, m, c)$  is false and requires QCDCL refutations of size  $2^{\Omega(n^\epsilon)}$ .

Our findings so far reveal an interesting picture on QCDCL hardness. Firstly, Proposition 11 and Corollary 22 imply that *not all Q-resolution hardness results lift to QCDCL*: the lower bounds for equality and random formulas shown via size-cost-capacity [8] *do*, but the lower bounds for parity shown via circuit complexity [12] *do not*.

Secondly, it is worth to compare the QCDCL hardness results for **Trapdoor** from the previous section to the QCDCL hardness results shown here for equality and random formulas. The hardness of **Trapdoor** lifts from propositional hardness for PHP, while the hardness of equality and random formulas lifts from Q-resolution hardness. In fact, this can be made formal by using a model of QBF proof systems with access to an NP oracle [17], which allows to collapse propositional subderivations of arbitrary size into just one oracle inference step. Hardness under the NP-oracle version of Q-resolution guarantees that the hardness is “genuine” to QBF and not lifted from propositional resolution. We show here that this notion of “genuine” QBF hardness, tailored towards QCDCL, also holds for the QCDCL lower bounds for equality and the random QBFs.

► **Proposition 23.** *The number of reduction steps in each long-distance QCDCL resolution refutation (and also each QCDCL<sup>LEV-ORD</sup><sub>RED</sub> refutation) of Equality <sub>$n$</sub>  is at least  $2^n$ . The analogous result holds for the false formulas  $Q(n, m, c)$  with  $2^{\Omega(n^\epsilon)}$  reduction steps.*

On the other hand, the parity formulas also exhibit “genuine” QBF hardness, as they are hard in the NP-oracle version of Q-resolution [10]. Since they are easy for QCDCL (Proposition 11), this means that not all genuine Q-resolution lower bounds lift to QCDCL.

Thirdly, hardness for QCDCL can of course also stem from hardness for long-distance Q-resolution, since the latter system p-simulates the former. However, there are only very few hardness results for long-distance Q-resolution known in the literature [3, 12, 13], hence

our hardness results shown here should be also valuable for practitioners, in particular the hardness results for the large class of random QCNFs. It is also worth noting that the equality formulas are easy for long-distance Q-resolution [9], hence our results imply an exponential separation between QCDCL and long-distance Q-resolution.

► **Proposition 24.** *Long-distance Q-resolution is exponentially stronger than QCDCL, i.e., long-distance Q-resolution  $p$ -simulates QCDCL and there are QCNFs that require exponential-size proofs in QCDCL, but admit polynomial-size proofs in long-distance Q-resolution.*

## 6 A QCDCL system that characterises Q-resolution

In one of our main results we obtain a QCDCL characterisation of Q-resolution. Of course, given that Q-resolution and QCDCL are incomparable (Section 4), we cannot hope to achieve such a characterisation by simply strengthening some of the QCDCL policies.<sup>6</sup> As explained in the previous section, traditional QCDCL is using the decision policy LEV-ORD and the unit-propagation policy RED. To obtain a QCDCL system equivalent to Q-resolution, we will have to change both policies. We will *strengthen* the decision policy and replace LEV-ORD by ANY-ORD (we could also replace it with the intermediate version ASS-ORD). In addition, we will somewhat *weaken* the unit propagation policy from RED to NO-RED.<sup>7</sup>

This leads to the following characterisation of Q-resolution.

► **Theorem 25.** *Q-resolution,  $\text{QCDCL}_{\text{NO-RED}}^{\text{ANY-ORD}}$ , and  $\text{QCDCL}_{\text{NO-RED}}^{\text{ASS-ORD}}$  are  $p$ -equivalent proof systems.*

*In particular, each Q-resolution refutation  $\pi$  of a QCNF in  $n$  variables can be transformed into a  $\text{QCDCL}_{\text{NO-RED}}^{\text{ASS-ORD}}$ -refutation of size  $\mathcal{O}(n^3 \cdot |\pi|)$  that uses an arbitrary asserting learning scheme.*

One part of the simulation above was already shown in Theorem 8, where we proved that all QCDCL systems with NO-RED are  $p$ -simulated by Q-resolution. The technically most challenging part is the reverse simulation where we need to construct  $\text{QCDCL}_{\text{NO-RED}}^{\text{ASS-ORD}}$  trails from Q-resolution proofs. The main conceptual notion we use is that of *reliable* clauses.

► **Definition 26.** *Let  $\Phi = \mathcal{Q} \cdot \phi$  be a QCNF and  $C$  be a non-tautological clause. If there is a  $\text{QCDCL}_{\text{NO-RED}}^{\text{ASS-ORD}}$  trail  $\mathcal{T}$ , an existential literal  $\ell \in C$  and a set of literals  $\alpha \subseteq \bar{C} \setminus \{\bar{\ell}\}$  such that  $\alpha$  is the set of decision literals in  $\mathcal{T}$  and  $\ell \in \mathcal{T}$ , then  $C$  is called *unreliable* with respect to  $\Phi$ . Alternatively, we say that the decisions  $\bar{C}$  are blocking each other.*

*If  $C$  is not unreliable, we call  $C$  reliable.*

Intuitively, a reliable clause  $C$  can be used to form a  $\text{QCDCL}_{\text{NO-RED}}^{\text{ASS-ORD}}$  trail by using all negated literals from  $C$  as decisions. This way we progress through the Q-resolution proof, successively learning clauses and making all clauses  $C$  in the Q-resolution proof unreliable until we obtain the empty clause.

This construction bears some similarities to the simulation of Q-resolution by CDCL [48], but poses further technical challenges due to quantification and the additional rules of Q-resolution. In the inductive argument for Theorem 25 we therefore need to distinguish three cases on whether  $C$  is an axiom or derived by resolution or reduction, each requiring its own lemma (Lemmas 27, 28, and 29). For the following lemmas, let  $\xi$  be an arbitrary, but fixed asserting learning scheme.

<sup>6</sup> Such hope might not have seemed totally implausible prior to this paper, e.g. [35] states that ‘CDCL QBF solving appears to be quite weak compared to general Q-resolution.’

<sup>7</sup> While intuitively NO-RED might indeed appear weaker than RED (it produces fewer unit propagations), we show in the next section that they are in fact incomparable, cf. Figure 2.



► **Lemma 27.** Let  $\Phi := \mathcal{Q} \cdot \phi$  be a QCNF in  $n$  variables and  $C \in \phi$ . If  $C$  is reliable with respect to  $\Phi$ , there exists a  $\text{QCDCL}_{\text{No-RED}}^{\text{Ass-ORD}}$ -proof  $\iota$  with trails  $\mathcal{T}_1, \dots, \mathcal{T}_{f_n}$  from  $\Phi$  of some clause  $E$  that uses the learning scheme  $\xi$  such that  $|\iota| \in \mathcal{O}(n^3)$ . If  $E \neq (\perp)$ , then  $C$  is unreliable with respect to  $\mathcal{Q} \cdot (\phi \cup \{\xi(\mathcal{T}_1), \dots, \xi(\mathcal{T}_{f_n})\})$ .

► **Lemma 28.** Let  $\Phi := \mathcal{Q} \cdot \phi$  be a QCNF in  $n$  variables. Also let  $C_1 \vee x$  be a clause that is unreliable with respect to  $\Psi := \mathcal{Q} \cdot \psi$  with  $\psi \subseteq \phi$  and  $C_2 \vee \bar{x}$  unreliable with respect to  $\Upsilon := \mathcal{Q} \cdot \tau$  with  $\tau \subseteq \phi$ , such that  $C_1 \vee C_2$  is non-tautological. Let  $\xi$  be an asserting learning scheme. If  $C_1 \vee C_2$  is reliable with respect to  $\Phi$ , there exists a  $\text{QCDCL}_{\text{No-RED}}^{\text{Ass-ORD}}$ -proof  $\iota$  with  $\theta(\iota) = \mathcal{T}_1, \dots, \mathcal{T}_{f_n}$  from  $\Phi$  of some clause  $E$  that uses the learning scheme  $\xi$  such that  $|\iota| \in \mathcal{O}(n^3)$ . If  $E \neq (\perp)$ , then  $C_1 \vee C_2$  is unreliable with respect to  $\mathcal{Q} \cdot (\phi \cup \{\xi(\mathcal{T}_1), \dots, \xi(\mathcal{T}_{f_n})\})$ .

► **Lemma 29.** Let  $\Phi := \mathcal{Q} \cdot \phi$  be a QCNF in  $n$  variables, let  $D := C \vee u_1 \vee \dots \vee u_m$  be a non-tautological clause with universal literals  $u_1, \dots, u_m$  and  $\text{red}(D) = C$ , such that  $D$  is unreliable with respect to a QCNF  $\Psi = \mathcal{Q} \cdot \psi$  with  $\psi \subseteq \phi$ . Let  $\xi$  be an asserting learning scheme. If  $C$  is reliable with respect to  $\Phi$ , there exists a  $\text{QCDCL}_{\text{No-RED}}^{\text{Ass-ORD}}$ -proof  $\iota$  with  $\theta(\iota) = \mathcal{T}_1, \dots, \mathcal{T}_{f_n}$  from  $\Phi$  of some clause  $E$  that uses the learning scheme  $\xi$  such that  $|\iota| \in \mathcal{O}(n^3)$ . If  $E \neq (\perp)$ , then  $C$  is unreliable with respect to  $\mathcal{Q} \cdot (\phi \cup \{\xi(\mathcal{T}_1), \dots, \xi(\mathcal{T}_{f_n})\})$ .

We also point out that in comparison to the notion of 1-empowering clauses from [48], our argument via reliability yields somewhat better bounds on the simulation, thereby implying a slight quantitative improvement by a factor of  $n$  in the simulation in [48]:

► **Theorem 30.** Let  $\phi$  be a CNF in  $n$  variables and let  $\pi$  be a resolution refutation of  $\phi$ . Then  $\phi$  has a CDCL refutation of size  $\mathcal{O}(n^3|\pi|)$ .

## 7 The simulation order of QCDCL proof systems

We can now analyse the simulation order of the defined QCDCL and QBF resolution systems, cf. Figure 2 which almost completely determines the simulations and separations between the systems involved (cf. Section 8 for the open cases).

We highlight the most interesting findings (in addition to the results already described).

Firstly, we show that the unit-propagation policies RED and NO-RED are incomparable when fixing the decision policy LEV-ORD used in practical QCDCL.

► **Theorem 31.** The systems  $\text{QCDCL}_{\text{No-RED}}^{\text{LEV-ORD}}$  and  $\text{QCDCL}_{\text{RED}}^{\text{LEV-ORD}}$  are incomparable.

For the separations we use the QBFs  $\text{QParity}_n$  and  $\text{Trapdoor}_n$ . For practice, this results means that it is a priori not clear that the unit-propagation policy as used in practical QCDCL is actually preferable to the simpler unit-propagation policy from CDCL (which would work in QCDCL as well).

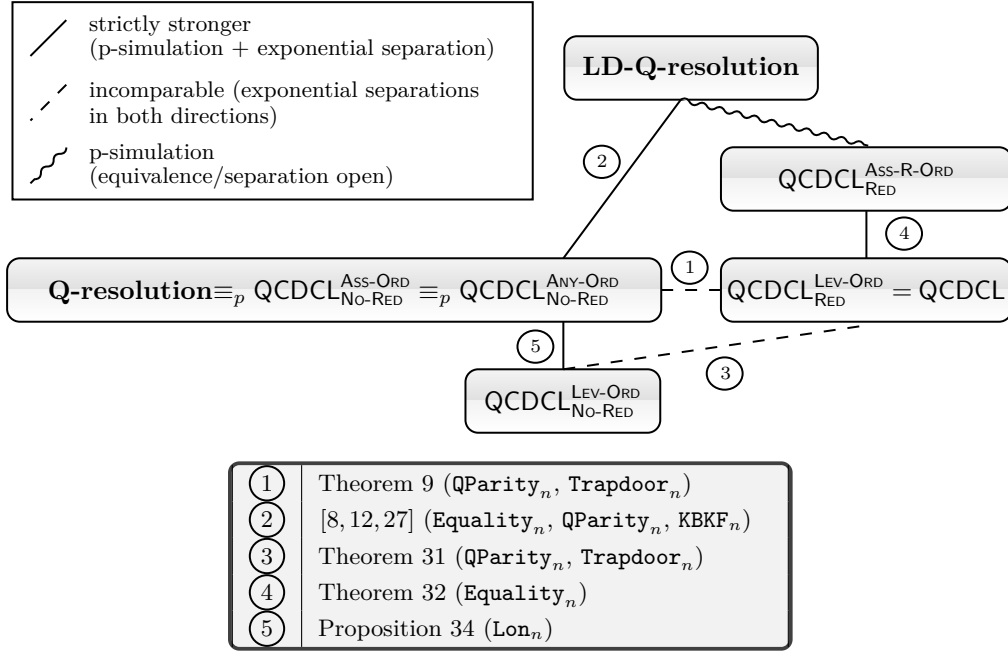
Secondly, we show that replacing the decision policy LEV-ORD in QCDCL with the more liberal decision policy ASS-R-ORD yields exponentially shorter QCDCL runs, which we demonstrate on the  $\text{Equality}_n$  formulas.

► **Theorem 32.**  $\text{QCDCL}_{\text{RED}}^{\text{ASS-R-ORD}}$  is exponentially stronger than  $\text{QCDCL}_{\text{RED}}^{\text{LEV-ORD}}$ .

Again, this theoretical result identifies potential for improvements in practical solving (cf. also the discussion in the concluding Section 8).

Thirdly, we recall the formulas  $\text{Lon}_n$  that were introduced by Lonsing in [39]. Originally, these QCNFs were constructed to separate QBF solvers that differ in the implemented dependency schemes (we do not consider these concepts here, though).





■ **Figure 2** The simulation order of QCDCL and QBF resolution systems. The table contains pointers to the separating formulas.

► **Definition 33** (Lonsing [39]). Let  $\text{Lon}_n$  be the QCNF

$$\exists a, b, b_1, \dots, b_{s_n} \forall x, y \exists c, d \cdot (a \vee x \vee c) \wedge (a \vee b \vee b_1 \vee \dots \vee b_{s_n}) \wedge (b \vee y \vee d) \wedge (x \vee c) \wedge (x \vee \bar{c}) \wedge \text{PHP}_n^{n+1}(b_1, \dots, b_{s_n}) .$$

It was shown in [39] that these formulas become easy to refute by choosing the standard dependency scheme. However,  $\text{Lon}_n$  serve as witnesses for separating our systems as well.

► **Proposition 34.** *The QCNFs  $\text{Lon}_n$  require exponential-size proofs in the proof systems  $\text{QCDCL}_{\text{RED}}^{\text{LEV-ORD}}$  and  $\text{QCDCL}_{\text{NO-RED}}^{\text{LEV-ORD}}$ , but have constant-size proofs in  $\text{QCDCL}_{\text{RED}}^{\text{ASS-R-ORD}}$  and Q-resolution.*

## 8 Conclusion

In this paper we performed a formal, proof-theoretic analysis of QCDCL. In particular, we focused on the relation of QCDCL and Q-resolution, showing both the incomparability of practically-used QCDCL to Q-resolution as well as the equivalence of a new QCDCL version to Q-resolution.

In addition to the theoretical contributions of this paper, we believe that our findings will also be interesting for practitioners. Firstly, because we have shown the first rigorous dedicated hardness results for QCDCL, not only in terms of formula families with at most one instance per input size (as is typical in proof complexity), but also in terms of a large family of random QBFs.

Secondly, we believe that it would be interesting to test the potential of our new QCDCL variants for practical solving. Though we have formulated these as proof systems, it should be fairly straightforward to incorporate our new policies into actual QCDCL implementations.

In particular, the insight that decisions do not need to follow the order of quantification in the prefix should be a welcome discovery. Of course, when just using the policy ANY-ORD, it is not clear that asserting clauses can always be learnt. Therefore, we suggest that for practical implementations, the most interesting new systems should be  $\text{QCDCL}_{\text{RED}}^{\text{ASS-ORD}}$  and  $\text{QCDCL}_{\text{RED}}^{\text{ASS-ORD}}$ . Both facilitate liberal decision policies, not necessarily following the prefix order, while still allowing to learn asserting clauses. Since both systems are incomparable, it is a priori not clear which one to prefer in practice. However, we would suggest that  $\text{QCDCL}_{\text{RED}}^{\text{ASS-ORD}}$  should be the more interesting system, since it uses the same unit propagation as QCDCL, but provides an exponential strengthening of QCDCL (as shown in Theorem 32) via the decision policy ASS-R-ORD.

We close with some open questions that are triggered by the results presented here:

- Can we find an alternative formula instead of  $\text{Trapdoor}_n$  for the separation between Q-resolution and QCDCL (easy for Q-resolution, hard for QCDCL)? I.e., we are primarily interested in formulas whose hardness does not depend on propositional resolution.
- Can we find a separation between  $\text{QCDCL}_{\text{RED}}^{\text{ASS-ORD}}$  and long-distance Q-resolution?
- Can we even find a separation between  $\text{QCDCL}_{\text{RED}}^{\text{ANY-ORD}}$  and long-distance Q-resolution, or are the systems possibly even equivalent?

---

## References

- 1 Albert Atserias, Johannes Klaus Fichte, and Marc Thurley. Clause-learning algorithms with many restarts and bounded-width resolution. *J. Artif. Intell. Res.*, 40:353–373, 2011.
- 2 Valeriy Balabanov and Jie-Hong R. Jiang. Unified QBF certification and its applications. *Form. Methods Syst. Des.*, 41(1):45–65, 2012.
- 3 Valeriy Balabanov, Magdalena Widl, and Jie-Hong R. Jiang. QBF resolution systems and their proof complexities. In *Proc. International Conference on Theory and Applications of Satisfiability Testing (SAT)*, pages 154–169, 2014.
- 4 Paul Beame, Henry A. Kautz, and Ashish Sabharwal. Towards understanding and harnessing the potential of clause learning. *J. Artif. Intell. Res. (JAIR)*, 22:319–351, 2004. doi:10.1613/jair.1410.
- 5 Paul Beame and Toniann Pitassi. Propositional proof complexity: Past, present, and future. In G. Paun, G. Rozenberg, and A. Salomaa, editors, *Current Trends in Theoretical Computer Science: Entering the 21st Century*, pages 42–70. World Scientific Publishing, 2001.
- 6 Olaf Beyersdorff and Joshua Blinkhorn. Lower bound techniques for QBF expansion. *Theory Comput. Syst.*, 64(3):400–421, 2020.
- 7 Olaf Beyersdorff, Joshua Blinkhorn, Leroy Chew, Renate A. Schmidt, and Martin Suda. Reinterpreting dependency schemes: Soundness meets incompleteness in DQBF. *J. Autom. Reason.*, 63(3):597–623, 2019.
- 8 Olaf Beyersdorff, Joshua Blinkhorn, and Luke Hinde. Size, cost, and capacity: A semantic technique for hard random QBFs. *Logical Methods in Computer Science*, 15(1), 2019.
- 9 Olaf Beyersdorff, Joshua Blinkhorn, and Meena Mahajan. Building strategies into QBF proofs. In *STACS, LIPIcs*, pages 14:1–14:18. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2019.
- 10 Olaf Beyersdorff, Joshua Blinkhorn, and Meena Mahajan. Hardness characterisations and size-width lower bounds for QBF resolution. In *Proc. ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 209–223. ACM, 2020.
- 11 Olaf Beyersdorff, Ilario Bonacina, Leroy Chew, and Jan Pich. Frege systems for quantified Boolean logic. *J. ACM*, 67(2), 2020.
- 12 Olaf Beyersdorff, Leroy Chew, and Mikolás Janota. New resolution-based QBF calculi and their proof complexity. *ACM Transactions on Computation Theory*, 11(4):26:1–26:42, 2019.

- 13 Olaf Beyersdorff, Leroy Chew, Meena Mahajan, and Anil Shukla. Feasible interpolation for QBF resolution calculi. *Logical Methods in Computer Science*, 13, 2017.
- 14 Olaf Beyersdorff, Leroy Chew, Meena Mahajan, and Anil Shukla. Understanding cutting planes for QBFs. *Inf. Comput.*, 262:141–161, 2018.
- 15 Olaf Beyersdorff, Leroy Chew, and Karteek Sreenivasaiah. A game characterisation of tree-like Q-Resolution size. *J. Comput. Syst. Sci.*, 104:82–101, 2019.
- 16 Olaf Beyersdorff and Luke Hinde. Characterising tree-like Frege proofs for QBF. *Inf. Comput.*, 268, 2019.
- 17 Olaf Beyersdorff, Luke Hinde, and Ján Pich. Reasons for hardness in QBF proof systems. *ACM Transactions on Computation Theory*, 12(2), 2020.
- 18 Olaf Beyersdorff, Mikolás Janota, Florian Lonsing, and Martina Seidl. Quantified Boolean formulas. In Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh, editors, *Handbook of Satisfiability, 2nd edition*, Frontiers in Artificial Intelligence and Applications. IOS press, 2021.
- 19 Maria Luisa Bonet, Sam Buss, and Jan Johannsen. Improved separations of regular resolution from clause learning proof systems. *J. Artif. Intell. Res.*, 49:669–703, 2014.
- 20 Samuel R. Buss. Towards NP-P via proof complexity and search. *Ann. Pure Appl. Logic*, 163(7):906–917, 2012.
- 21 Samuel R. Buss, Jan Hoffmann, and Jan Johannsen. Resolution trees with lemmas: Resolution refinements that characterize DLL algorithms with clause learning. *Logical Methods in Computer Science*, 4(4), 2008. doi:10.2168/LMCS-4(4:13)2008.
- 22 Marco Cadoli, Andrea Giovanardi, and Marco Schaerf. An algorithm to evaluate quantified Boolean formulae. In *Proc. AAAI Conference on Artificial Intelligence (AAAI)*, pages 262–267, 1998.
- 23 Stephen A. Cook and Tsuyoshi Morioka. Quantified propositional calculus and a second-order theory for NC<sup>1</sup>. *Arch. Math. Log.*, 44(6):711–749, 2005.
- 24 Stephen A. Cook and Robert A. Reckhow. The relative efficiency of propositional proof systems. *The Journal of Symbolic Logic*, 44(1):36–50, 1979.
- 25 Martin Davis, George Logemann, and Donald W. Loveland. A machine program for theorem-proving. *Commun. ACM*, 5(7):394–397, 1962. doi:10.1145/368273.368557.
- 26 Martin Davis and Hilary Putnam. A computing procedure for quantification theory. *Journal of the ACM*, 7:210–215, 1960.
- 27 Uwe Egly, Florian Lonsing, and Magdalena Widl. Long-distance resolution: Proof generation and strategy extraction in search-based QBF solving. In *Proc. Logic for Programming, Artificial Intelligence, and Reasoning (LPAR)*, pages 291–308, 2013.
- 28 Peter Faymonville, Bernd Finkbeiner, Markus N. Rabe, and Leander Tentrup. Encodings of bounded synthesis. In *Proc. Tools and Algorithms for the Construction and Analysis of Systems - 23rd International Conference (TACAS)*, pages 354–370, 2017.
- 29 Merrick L. Furst, James B. Saxe, and Michael Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical Systems Theory*, 17(1):13–27, 1984.
- 30 Enrico Giunchiglia, Paolo Marin, and Massimo Narizzano. Reasoning with quantified Boolean formulas. In Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh, editors, *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*, pages 761–780. IOS Press, 2009.
- 31 Amin Haken. The intractability of resolution. *Theoretical Computer Science*, 39:297–308, 1985.
- 32 Johan Håstad. *Computational Limitations of Small Depth Circuits*. MIT Press, Cambridge, 1987.
- 33 Philipp Hertel, Fahiem Bacchus, Toniann Pitassi, and Allen Van Gelder. Clause learning can effectively p-simulate general propositional resolution. In *Proc. AAAI Conference on Artificial Intelligence (AAAI)*, pages 283–290, 2008. URL: <http://www.aaai.org/Library/AAAI/2008/aaai08-045.php>.

- 34 Marijn J. H. Heule, Martina Seidl, and Armin Biere. Solution validation and extraction for QBF preprocessing. *J. Autom. Reason.*, 58(1):97–125, 2017.
- 35 Mikolás Janota. On Q-resolution and CDCL QBF solving. In *Proc. International Conference on Theory and Applications of Satisfiability Testing (SAT’16)*, pages 402–418, 2016.
- 36 Mikolás Janota and Joao Marques-Silva. Expansion-based QBF solving versus Q-resolution. *Theor. Comput. Sci.*, 577:25–42, 2015.
- 37 Hans Kleine Büning, Marek Karpinski, and Andreas Flögel. Resolution for quantified Boolean formulas. *Inf. Comput.*, 117(1):12–18, 1995.
- 38 Jan Krajčček. *Proof complexity*, volume 170 of *Encyclopedia of Mathematics and Its Applications*. Cambridge University Press, 2019.
- 39 Florian Lonsing. *Dependency Schemes and Search-Based QBF Solving: Theory and Practice*. PhD thesis, Johannes Kepler University Linz, 2012.
- 40 Florian Lonsing and Uwe Egly. DepQBF 6.0: A search-based QBF solver beyond traditional QCDCL. In *Proc. International Conference on Automated Deduction (CADE)*, pages 371–384, 2017.
- 41 Florian Lonsing, Uwe Egly, and Allen Van Gelder. Efficient clause learning for quantified Boolean formulas via QBF pseudo unit propagation. In *Proc. International Conference on Theory and Applications of Satisfiability Testing (SAT)*, pages 100–115, 2013.
- 42 João P. Marques Silva, Inês Lynce, and Sharad Malik. Conflict-driven clause learning SAT solvers. In *Handbook of Satisfiability*. IOS Press, 2009.
- 43 João P. Marques Silva and Karem A. Sakallah. GRASP - a new search algorithm for satisfiability. In *ICCAD*, pages 220–227, 1996. doi:10.1145/244522.244560.
- 44 Nathan Mull, Shuo Pang, and Alexander A. Razborov. On CDCL-based proof systems with the ordered decision strategy. In *Proc. International Conference on Theory and Applications of Satisfiability Testing (SAT’20)*, pages 149–165. Springer, 2020.
- 45 Jakob Nordström. *Short Proofs May Be Spacious : Understanding Space in Resolution*. PhD thesis, Royal Institute of Technology, Stockholm, Sweden, 2008.
- 46 Jakob Nordström. On the interplay between proof complexity and SAT solving. *SIGLOG News*, 2(3):19–44, 2015.
- 47 Tomás Peitl, Friedrich Slivovsky, and Stefan Szeider. Dependency learning for QBF. *J. Artif. Intell. Res.*, 65:180–208, 2019.
- 48 Knot Pipatsrisawat and Adnan Darwiche. On the power of clause-learning SAT solvers as resolution engines. *Artif. Intell.*, 175(2):512–525, 2011. doi:10.1016/j.artint.2010.10.002.
- 49 Luca Pulina and Martina Seidl. The 2016 and 2017 QBF solvers evaluations (QBFEVAL’16 and QBFEVAL’17). *Artif. Intell.*, 274:224–248, 2019.
- 50 Nathan Segerlind. The complexity of propositional proofs. *Bulletin of Symbolic Logic*, 13(4):417–481, 2007.
- 51 Ankit Shukla, Armin Biere, Luca Pulina, and Martina Seidl. A survey on applications of quantified Boolean formulas. In *Proc. IEEE International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 78–84, 2019.
- 52 Friedrich Slivovsky and Stefan Szeider. Soundness of Q-resolution with dependency schemes. *Theoretical Computer Science*, 612:83–101, 2016.
- 53 Moshe Y. Vardi. Boolean satisfiability: theory and engineering. *Commun. ACM*, 57(3):5, 2014.
- 54 Marc Vinyals. Hard examples for common variable decision heuristics. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2020.
- 55 Lintao Zhang, Conor F. Madigan, Matthew W. Moskewicz, and Sharad Malik. Efficient conflict driven learning in Boolean satisfiability solver. In *Proc. IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 279–285, 2001.
- 56 Lintao Zhang and Sharad Malik. Conflict driven learning in a quantified Boolean satisfiability solver. In *Proc. IEEE/ACM International Conference on Computer-aided Design (ICCAD)*, pages 442–449, 2002.

# The Quantum Supremacy Tsirelson Inequality

William Kretschmer 

University of Texas at Austin, TX, USA

<https://www.cs.utexas.edu/~kretsch/>

kretsch@cs.utexas.edu

---

## Abstract

A leading proposal for verifying near-term quantum supremacy experiments on noisy random quantum circuits is linear cross-entropy benchmarking. For a quantum circuit  $C$  on  $n$  qubits and a sample  $z \in \{0, 1\}^n$ , the benchmark involves computing  $|\langle z|C|0^n \rangle|^2$ , i.e. the probability of measuring  $z$  from the output distribution of  $C$  on the all zeros input. Under a strong conjecture about the classical hardness of estimating output probabilities of quantum circuits, no polynomial-time classical algorithm given  $C$  can output a string  $z$  such that  $|\langle z|C|0^n \rangle|^2$  is substantially larger than  $\frac{1}{2^n}$  (Aaronson and Gunn, 2019). On the other hand, for a random quantum circuit  $C$ , sampling  $z$  from the output distribution of  $C$  achieves  $|\langle z|C|0^n \rangle|^2 \approx \frac{2}{2^n}$  on average (Arute et al., 2019).

In analogy with the Tsirelson inequality from quantum nonlocal correlations, we ask: can a polynomial-time quantum algorithm do substantially better than  $\frac{2}{2^n}$ ? We study this question in the query (or black box) model, where the quantum algorithm is given oracle access to  $C$ . We show that, for any  $\varepsilon \geq \frac{1}{\text{poly}(n)}$ , outputting a sample  $z$  such that  $|\langle z|C|0^n \rangle|^2 \geq \frac{2+\varepsilon}{2^n}$  on average requires at least  $\Omega\left(\frac{2^{n/4}}{\text{poly}(n)}\right)$  queries to  $C$ , but not more than  $O(2^{n/3})$  queries to  $C$ , if  $C$  is either a Haar-random  $n$ -qubit unitary, or a canonical state preparation oracle for a Haar-random  $n$ -qubit state. We also show that when  $C$  samples from the Fourier distribution of a random Boolean function, the naive algorithm that samples from  $C$  is the optimal 1-query algorithm for maximizing  $|\langle z|C|0^n \rangle|^2$  on average.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Quantum complexity theory; Theory of computation  $\rightarrow$  Oracles and decision trees

**Keywords and phrases** quantum supremacy, quantum query complexity, random circuit sampling

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.13

**Related Version** Full version available at <https://arxiv.org/abs/2008.08721>.

**Funding** William Kretschmer: Supported by a Vannevar Bush Fellowship and a National Defense Science and Engineering Graduate (NDSEG) Fellowship from the US Department of Defense.

**Acknowledgements** Thanks to Scott Aaronson, Sabee Grewal, Sam Gunn, Robin Kothari, Daniel Liang, Patrick Rall, Andrea Rocchetto, and Justin Thaler for helpful discussions and illuminating insights. Thanks also to anonymous reviewers for helpful comments regarding the presentation of this work.

## 1 Introduction

A team based at Google has claimed the first experimental demonstration of quantum computational supremacy on a programmable device [9]. The experiment involved *random circuit sampling*, where the task is to sample (with nontrivial fidelity) from the output distribution of a quantum circuit containing random 1- and 2-qubit gates. To verify their experiment, they used the so-called *Linear Cross-Entropy Benchmark*, or Linear XEB. Specifically, for an  $n$ -qubit quantum circuit  $C$  and samples  $z_1, \dots, z_k \in \{0, 1\}^n$ , the benchmark is given by:

$$b = \frac{2^n}{k} \cdot \sum_{i=1}^k |\langle z_i|C|0^n \rangle|^2.$$



© William Kretschmer;

licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 13; pp. 13:1–13:13

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

## 13:2 The Quantum Supremacy Tsirelson Inequality

The goal is for  $b$  to be large with high probability over the choice of the random circuit and the randomness of the sampler, as this demonstrates that the observations tend to concentrate on the outputs that are more likely to be measured under the ideal distribution for  $C$  (i.e. the noiseless distribution in which  $z$  is measured with probability  $|\langle z|C|0^n \rangle|^2$ ). We formalize this task as the  $b$ -XHOG task:

► **Problem 1** ( $b$ -XHOG, or Linear Cross-Entropy Heavy Output Generation). *Given a quantum circuit  $C$  on  $n$  qubits, output a sample  $z \in \{0, 1\}^n$  such that  $\mathbb{E} [|\langle z|C|0^n \rangle|^2] \geq \frac{b}{2^n}$ , where the expectation is over an implicit distribution over circuits  $C$  and over the randomness of the algorithm that outputs  $z$ .*

Here,  $b$  “large” means  $b$  bounded away from 1, as outputting  $z$  uniformly at random achieves  $b = 1$  on average for any  $C$ . On the other hand, if  $z$  is drawn from the ideal noiseless distribution for  $C$ , and if the random circuits  $C$  empirically exhibit the *Porter-Thomas* distribution on output probabilities, then sampling from  $C$  achieves  $b \approx 2$  [9, 2].

Under a strong complexity-theoretic conjecture about the classical hardness of nontrivially estimating output probabilities of quantum circuits, Aaronson and Gunn showed that no classical polynomial-time algorithm can solve  $b$ -XHOG for any  $b \geq 1 + \frac{1}{\text{poly}(n)}$  on random quantum circuits of polynomial size [2]. Thus, a physical quantum computer that solves  $b$ -XHOG for  $b \geq 1 + \Omega(1)$  is considered strong evidence of quantum computational supremacy.

In this work, we ask: can an efficient quantum algorithm for  $b$ -XHOG do substantially better than  $b = 2$ ? That is, what is the largest  $b$  for which a polynomial-time quantum algorithm can solve  $b$ -XHOG on random circuits? Note that the largest  $b$  we could hope for is achieved by the optimal sampler that always outputs the string  $z$  maximizing  $|\langle z|C|0^n \rangle|^2$ . If the random circuits induce a Porter-Thomas distribution on output probabilities, then this solves  $b$ -XHOG for  $b = \Theta(n)$ , because the probabilities of a Porter-Thomas distribution approach i.i.d. exponential random variables (see Fact 10 below). However, finding the largest output probability might be computationally difficult even on a quantum computer, which is why we restrict our attention to *efficient* quantum algorithms.

We refer to our problem as the “quantum supremacy Tsirelson inequality” in reference to the Bell [11] and Tsirelson [18] inequalities for quantum nonlocal correlations (for a modern overview, see [20]). Under this analogy, the quantity  $b$  in XHOG plays a similar role as the probability  $p$  of winning some nonlocal game. For example, the Bell inequality for the CHSH game [19] states that no classical strategy can win the game with probability  $p > \frac{3}{4}$ ; we view this as analogous to the conjectured inability of efficient classical algorithms to solve  $b$ -XHOG for any  $b > 1$ . By contrast, a quantum strategy with pre-shared entanglement allows players to win the CHSH game with probability  $p = \cos^2(\frac{\pi}{8}) \approx 0.854 > \frac{3}{4}$ . An experiment that wins the CHSH game with probability  $p > \frac{3}{4}$ , a violation of the Bell inequality, is analogous to an experimental demonstration of  $b$ -XHOG for  $b > 1$  on a quantum computer that establishes quantum computational supremacy. Finally, the Tsirelson inequality for the CHSH game states that any quantum strategy involving arbitrary pre-shared entanglement wins with probability  $p \leq \cos^2(\frac{\pi}{8})$ . Hence, an upper bound on  $b$  for efficient quantum algorithms is the quantum supremacy counterpart to the Tsirelson inequality. We emphasize that our choice to refer to this as a “Tsirelson inequality” is purely by analogy; we do not claim that the question involving quantum supremacy or the techniques one might use to answer it are otherwise related to quantum nonlocal correlations.

### 1.1 Our Results

We study the quantum supremacy Tsirelson inequality in the quantum query (or black box) model. That is, we consider distributions over quantum circuits that make queries to a randomized quantum or classical oracle, and ask how many queries to the oracle are needed



to solve  $b$ -XHOG, in terms of  $b$ . Our motivation for studying this problem in the query model is twofold. First, quantum query results often give useful intuition for what to expect in the real world, and can provide insight into why naive algorithmic approaches fail. Second, we view this as an interesting quantum query complexity problem in its own right. Whereas most other quantum query lower bounds involve decision problems [5] or relation problems [12], XHOG is more like a weighted, average-case relation problem, because we only require that  $|\langle z|C|0^n\rangle|^2$  be large *on average*. Contrast this with the relation problem considered in [1], where the task is to output a  $z$  such that  $|\langle z|C|0^n\rangle|^2$  is greater than some threshold.

Note that there are known quantum query complexity lower bounds for relation problems [9], and even relation problems where the output is a quantum state [6, 25]. Yet, it is unclear whether existing quantum query lower bound techniques are useful here. Whereas the adversary method tightly characterizes the quantum query complexity of decision problems and state conversion problems [24], it is not even known to characterize the query complexity of relation problems (unless they are efficiently verifiable) [12]. The adversary method appears to be essentially useless for saying anything about XHOG, which is not efficiently verifiable and is not a relation problem in the traditional sense.<sup>1</sup>

The XHOG task is well-defined for any distribution of random quantum circuits, so this gives us a choice in selecting the distribution. We focus on three classes of oracle circuits that either resemble random circuits used in practical experiments, or that were previously studied in the context of quantum supremacy.

### Canonical State Preparation Oracles

Because the linear cross-entropy benchmark for a circuit  $C$  depends only on the state  $|\psi\rangle := C|0^n\rangle$  produced by the circuit on the all zeros input, it is natural to consider an oracle  $\mathcal{O}_\psi$  that prepares a random state  $|\psi\rangle$  without leaking additional information about  $|\psi\rangle$ . Formally, we choose a Haar-random  $n$ -qubit state  $|\psi\rangle$ , and fix a canonical state  $|\perp\rangle$  orthogonal to all  $n$ -qubit states.<sup>2</sup> Then, we take the oracle  $\mathcal{O}_\psi$  that acts as  $\mathcal{O}_\psi|\perp\rangle = |\psi\rangle$ ,  $\mathcal{O}_\psi|\psi\rangle = |\perp\rangle$ , and  $\mathcal{O}_\psi|\varphi\rangle = |\varphi\rangle$  for any state  $|\varphi\rangle$  that is orthogonal to both  $|\perp\rangle$  and  $|\psi\rangle$ . Equivalently,  $\mathcal{O}_\psi$  is the reflection about the state  $\frac{|\psi\rangle - |\perp\rangle}{2}$ . Finally, we let  $C$  be the composition of  $\mathcal{O}_\psi$  with any unitary that sends  $|0^n\rangle$  to  $|\perp\rangle$ , so that  $C|0^n\rangle = |\psi\rangle$ . This model is often chosen when proving lower bounds for quantum algorithms that query state preparation oracles (see e.g. [7, 3, 13]), in part because the ability to simulate  $\mathcal{O}_\psi$  follows in a completely black box manner from the ability to prepare  $|\psi\rangle$  unitarily without garbage (see Lemma 7 below). Hence, the oracle  $\mathcal{O}_\psi$  is “canonical” in the sense that it is uniquely determined by  $|\psi\rangle$  and is not any more powerful than any other oracle that prepares  $|\psi\rangle$  without garbage.

### Haar-Random Unitaries

A random polynomial-size quantum circuit  $C$  does not behave like a canonical state preparation oracle:  $C|x\rangle$  looks like a random quantum state for *any* computational basis state  $|x\rangle$ , not just  $x = 0^n$ . Indeed, random quantum circuits are known to information-theoretically approximate the Haar measure in certain regimes [14, 21], and it seems plausible that they are also computationally difficult to distinguish from the Haar measure. Thus, one could alternatively model random quantum circuits by Haar-random  $n$ -qubit unitaries.

<sup>1</sup> As we will see later, however, the polynomial method [10] plays an important role in one of our results.

<sup>2</sup> We can always assume that a convenient  $|\perp\rangle$  exists by extending the Hilbert space, if needed. For example, if  $|\psi\rangle$  is an  $n$ -qubit state, a natural choice is to encode  $|\psi\rangle$  by  $|\psi\rangle|1\rangle$  and to choose  $|\perp\rangle = |0^n\rangle|0\rangle$ .



### Fourier Sampling Circuits

Finally, we consider quantum circuits that query a random *classical* oracle. For this, we use FOURIER SAMPLING circuits, which Aaronson and Chen [1] previously studied in the context of proving oracular quantum supremacy for a problem related to XHOG. FOURIER SAMPLING circuits are defined as  $H^{\otimes n} U_f H^{\otimes n}$ , where  $U_f$  is a phase oracle for a uniformly random Boolean function  $f : \{0, 1\}^n \rightarrow \{-1, 1\}$ . On the all-zeros input, FOURIER SAMPLING circuits output a string  $z \in \{0, 1\}^n$  with probability proportional to the squared Fourier coefficient  $\hat{f}(z)^2$ . This model has the advantage that in principle, one can prove the corresponding quantum supremacy Bell inequality for classical algorithms given query access to  $f$ , and that in some cases one can replace  $f$  by a pseudorandom function to base quantum supremacy on cryptographic assumptions [1].

### Summary of Results

Our first result is an exponential lower bound on the number of quantum queries needed to solve  $(2 + \varepsilon)$ -XHOG given either of the two types of quantum oracles that we consider:

► **Theorem 2** (Informal version of Theorem 14 and Theorem 17). *For any  $\varepsilon \geq \frac{1}{\text{poly}(n)}$ , any quantum query algorithm for  $(2 + \varepsilon)$ -XHOG with query access to either:*

- (1) *a canonical state preparation oracle  $\mathcal{O}_\psi$  for a Haar-random  $n$ -qubit state  $|\psi\rangle$ , or*
- (2) *a Haar-random  $n$ -qubit unitary,*

*requires at least  $\Omega\left(\frac{2^{n/4}}{\text{poly}(n)}\right)$  queries.*

We do not know if Theorem 2 is optimal, but we show in Theorem 15 that a simple algorithm based on the quantum collision finding algorithm [16] solves  $(2 + \Omega(1))$ -XHOG using  $O(2^{n/3})$  queries to either oracle.

Finally, we show that for FOURIER SAMPLING circuits, the naive algorithm of simply running the circuit is optimal among all 1-query algorithms:

► **Theorem 3** (Informal version of Theorem 19). *Any 1-query quantum algorithm for  $b$ -XHOG with FOURIER SAMPLING circuits achieves  $b \leq 3$ .<sup>3</sup>*

## 1.2 Our Techniques

The starting point for our proof of the Tsirelson inequality with a canonical state preparation oracle  $\mathcal{O}_\psi$  is a result of Ambainis, Rosmanis, and Unruh [7], which shows that any algorithm that queries  $\mathcal{O}_\psi$  can be approximately simulated by a different algorithm that makes no queries, but starts with copies of a resource state that depends on  $|\psi\rangle$ . This resource state consists of polynomially many (in the number of queries to  $\mathcal{O}_\psi$ ) states of the form  $\alpha|\psi\rangle + \beta|\perp\rangle$ , i.e. copies of  $|\psi\rangle$  in superposition with  $|\perp\rangle$ . Our strategy is to show that if any algorithm solves  $b$ -XHOG given this resource state, then a similar algorithm solves  $b$ -XHOG given copies of  $|\psi\rangle$  alone. Then, we prove a lower bound on the number of copies of  $|\psi\rangle$  needed to solve  $b$ -XHOG. To do so, we argue that if  $|\psi\rangle$  is Haar-random, then the best algorithm

<sup>3</sup> Note that the value of  $b$  achieved by the naive quantum algorithm for XHOG depends on the class of circuits used. In contrast to Haar-random circuits that achieve  $b \approx 2$ , FOURIER SAMPLING circuits achieve  $b \approx 3$  (see Proposition 18). This stems from the fact that the amplitudes of a Haar-random quantum state are approximately distributed as *complex* normal random variables, whereas the amplitudes of a state produced by a random FOURIER SAMPLING circuit are approximately distributed as *real* normal random variables.

for  $b$ -XHOG given copies of  $|\psi\rangle$  is a simple collision-finding algorithm: measure all copies of  $|\psi\rangle$  in the computational basis, and output whichever string  $z \in \{0,1\}^n$  appears most frequently in the measurement results. For a Haar-random  $n$ -qubit state, the chance of seeing any collisions is exponentially unlikely (unless the number of copies of  $|\psi\rangle$  is exponentially large in  $n$ ), and so this does not do much better than measuring a single copy of  $|\psi\rangle$  and outputting the result.

To prove the analogous lower bound for  $b$ -XHOG with a Haar-random unitary oracle, we show more generally that the canonical state preparation oracles and Haar-random unitary oracles are essentially equivalent as resources, which may be of independent interest. More specifically, we show that for an  $n$ -qubit state  $|\psi\rangle$ , given query access to  $\mathcal{O}_\psi$ , one can approximately simulate (to exponential precision) a random oracle that prepares  $|\psi\rangle$ . By “random oracle that prepares  $|\psi\rangle$ ,” we mean an  $n$ -qubit unitary  $U_\psi$  that acts as  $U_\psi|0^n\rangle = |\psi\rangle$  but Haar-random everywhere else. We can construct such a  $U_\psi$  by taking an arbitrary  $n$ -qubit unitary that maps  $|0^n\rangle$  to  $|\psi\rangle$ , then composing it with a Haar-random unitary on the  $(2^n - 1)$ -dimensional subspace orthogonal to  $|0^n\rangle$ .

Our lower bound for FOURIER SAMPLING circuits uses an entirely different technique. We use the polynomial method of Beals et al. [10], which shows that for any quantum algorithm that makes  $T$  queries to a classical oracle, the output probabilities of the algorithm can be expressed as degree- $2T$  polynomials in the variables of the classical oracle. Our key observation is that the average linear XEB score achieved by such a quantum query algorithm can *also* be expressed as a polynomial in the variables of the classical oracle. We further observe that this polynomial is constrained by the requirement that the polynomials representing the output probabilities must be nonnegative and sum to 1. This allows us to upper bound the largest linear XEB score achievable by the maximum value of a certain linear program, whose variables are the coefficients of the polynomials that represent the output probabilities of the algorithm. To upper bound this quantity, we exhibit a solution to the dual linear program.

Due to space constraints, we defer the proofs to the full version of this paper, available at <https://arxiv.org/abs/2008.08721>.

## 2 Preliminaries

### 2.1 Notation

We use  $[N]$  to denote the set  $\{1, 2, \dots, N\}$ . We use  $\mathbb{1}$  to denote the identity matrix (of implicit size). We let  $\text{TD}(\rho, \sigma)$  denote the trace distance between density matrices  $\rho$  and  $\sigma$ , and let  $\|A\|_\diamond$  denote the diamond norm of a superoperator  $A$  acting on density matrices (see [4] for definitions). For a unitary matrix  $U$ , we use  $U \cdot U^\dagger$  to denote the superoperator that maps  $\rho$  to  $U\rho U^\dagger$ . In a slight abuse of notation, if  $A$  denotes a quantum algorithm (which may consist of unitary gates, measurements, oracle queries, and initialization of ancilla qubits), then we also use  $A$  to denote the superoperator corresponding to the action of  $A$  on input density matrices.

### 2.2 Oracles for Quantum States

We frequently consider quantum algorithms that query quantum oracles. In this model, a query to a unitary matrix  $U$  consists of a single application of either  $U$ ,  $U^\dagger$ , or controlled versions of  $U$  or  $U^\dagger$ . We also consider quantum algorithms that make queries to *random* oracles. In analogue with the classical random oracle model, such calls are not randomized

at each query. Rather, a unitary  $U$  is chosen randomly (from some distribution) at the start of the execution of the algorithm, and thereafter all queries for the duration of the algorithm are made to  $U$ .

We now define several types of unitary oracles that we will use. These definitions (and associated lemmas giving constructions of them) have appeared implicitly or explicitly in prior work, e.g. [7, 3, 13, 8]. For completeness, we provide proofs of the constructions in the full version.

► **Definition 4.** For an  $n$ -qubit quantum state  $|\psi\rangle$ , the reflection about  $|\psi\rangle$ , denoted  $\mathcal{R}_\psi$ , is the  $n$ -qubit unitary  $\mathcal{R}_\psi := \mathbb{1} - 2|\psi\rangle\langle\psi|$ .

In other words,  $|\psi\rangle$  is a  $-1$  eigenstate of  $\mathcal{R}_\psi$ , and all states orthogonal to  $|\psi\rangle$  are  $+1$  eigenstates. Note that some authors define the reflection about  $|\psi\rangle$  to be the negation of this operator (e.g. [26, 28, 8]), while others follow our convention (e.g. [15, 23, 3]). This makes little difference, as these definitions are equivalent up to a global phase (or, if using the controlled versions, equivalent up to a Pauli  $Z$  gate).

The following lemma shows that  $\mathcal{R}_\psi$  can be simulated given any unitary that prepares  $|\psi\rangle$  from the all-zeros state, possibly with unentangled garbage.

► **Lemma 5.** Let  $U$  be a unitary that acts as  $U|0^n\rangle|0^m\rangle = |\psi\rangle|\varphi\rangle$ , where  $|\psi\rangle$  and  $|\varphi\rangle$  are  $n$ - and  $m$ -qubit states, respectively. Then one can simulate  $T$  queries to the reflection  $\mathcal{R}_\psi$  using  $2T + 1$  queries to  $U$ .

► **Definition 6.** For a quantum state  $|\psi\rangle$ , the canonical state preparation oracle for  $|\psi\rangle$ , denoted  $\mathcal{O}_\psi$ , is the reflection about the state  $\frac{|\psi\rangle - |\perp\rangle}{\sqrt{2}}$ , where  $|\perp\rangle$  is some canonical state orthogonal to  $|\psi\rangle$ .

Unless otherwise specified, we generally assume that if  $|\psi\rangle$  is an  $n$ -qubit state, then  $|\perp\rangle$  is orthogonal to the space of  $n$ -qubit states under a suitable encoding (see Footnote 2).

The next lemma shows that  $\mathcal{O}_\psi$  can be simulated from *any* oracle that prepares  $|\psi\rangle$  without garbage:

► **Lemma 7.** Let  $U$  be an  $n$ -qubit unitary that satisfies  $U|0^n\rangle = |\psi\rangle$ . Then one can simulate  $T$  queries to  $\mathcal{O}_\psi$  using  $4T + 2$  queries to  $U$ .

We introduce the notion of a *random* state preparation oracle, which, to our knowledge, is new.

► **Definition 8.** For an  $n$ -qubit state  $|\psi\rangle$  we define a random state preparation oracle for  $|\psi\rangle$ , denoted  $U_\psi$ , as follows. We fix an arbitrary  $n$ -qubit unitary  $V$  that satisfies  $V|0^n\rangle = |\psi\rangle$ , then choose a Haar-random unitary  $W$  that acts on the  $(2^n - 1)$ -dimensional subspace orthogonal to  $|0^n\rangle$  in the space of  $n$ -qubit states. Finally, we set  $U_\psi = VW$ .

The invariance of the Haar measure guarantees that this distribution over  $U_\psi$  is independent of the choice of  $V$ , and hence this is well-defined. Note that while we often refer to  $U_\psi$  as a single unitary matrix,  $U_\psi$  really refers to a *distribution* over unitary matrices. Notice also that if  $|\psi\rangle$  is distributed as a Haar-random  $n$ -qubit state, then  $U_\psi$  is distributed as a Haar-random  $n$ -qubit unitary.

## 2.3 Other Useful Facts

We use the following formula for the distance between unitary superoperators in the diamond norm.

► **Fact 9** ([4]). Let  $V$  and  $W$  be unitary matrices, and suppose  $d$  is the distance between 0 and the polygon in the complex plane whose vertices are the eigenvalues of  $VW^\dagger$ . Then

$$\|V \cdot V^\dagger - W \cdot W^\dagger\|_\diamond = 2\sqrt{1-d^2}.$$

Finally, we observe that for a Haar-random  $n$ -qubit quantum state, the information-theoretically largest linear XEB achievable is  $O(n)$ .

► **Fact 10.** Let  $|\psi\rangle$  be a Haar-random  $n$ -qubit quantum state. Then:

$$\mathbb{E}_{|\psi\rangle} \left[ \max_{z \in \{0,1\}^n} |\langle z|\psi\rangle|^2 \right] \leq \frac{O(n)}{2^n}.$$

### 3 Canonical State Preparation Oracles

In this section, we prove the quantum supremacy Tsirelson inequality for XHOG with a canonical state preparation oracle for a Haar-random state. We first sketch the important ideas in the proof. At the heart of our proof is the following lemma, due to Ambainis, Rosmanis, and Unruh [7]. It shows that any quantum algorithm that makes queries to a canonical state preparation oracle  $\mathcal{O}_\psi$  can be approximately simulated by a quantum algorithm that makes no queries to  $\mathcal{O}_\psi$ , and instead receives various copies of  $|\psi\rangle$  and superpositions of  $|\psi\rangle$  with some canonical orthogonal state.

► **Lemma 11** ([7]). Let  $A$  be a quantum query algorithm that makes  $T$  queries to  $\mathcal{O}_\psi$ . Then for any  $k$ , there is a quantum algorithm  $B$  that makes no queries to  $\mathcal{O}_\psi$ , and a quantum state  $|R\rangle$  of the form:

$$|R\rangle := \bigotimes_{j=1}^k \alpha_j |\psi\rangle + \beta_j |\perp\rangle$$

such that for any state  $|\varphi\rangle$ :

$$\text{TD}(A(|\varphi\rangle\langle\varphi|), B(|R\rangle\langle R|, |\varphi\rangle\langle\varphi|)) \leq O\left(\frac{T}{\sqrt{k}}\right).$$

So long as  $k \gg T^2$ , the output of  $B$  will be arbitrarily close to the output of  $A$  in trace distance. We will use this and Fact 10 to show that if  $A$  solves  $b$ -XHOG for some  $b > 2$ , then so does  $B$ . Then, to prove a lower bound on the number of queries  $T$  to  $\mathcal{O}_\psi$  needed to solve  $b$ -XHOG, it suffices to instead lower bound  $k$ , the number of states of the form  $\alpha_j |\psi\rangle + \beta_j |\perp\rangle$  needed to solve  $b$ -XHOG.

When  $|\psi\rangle$  is a Haar-random state, notice that the linear XEB depends only on the *magnitude* of the amplitudes in  $|\psi\rangle$ ; the phases are irrelevant. So, when considering algorithms that attempt to solve  $b$ -XHOG given only a state  $|R\rangle$  of the form used in Lemma 11, we might as well assume that the algorithm randomly reassigns the phases on  $|\psi\rangle$ . More formally, define the mixed state  $\sigma_R$  as

$$\sigma_R := \mathbb{E}_{\text{diagonal } U} [U^{\otimes k} |R\rangle\langle R| U^{\dagger \otimes k}], \quad (1)$$

where the expectation is over the diagonal unitaries  $U$  such that the entries  $\langle i|U|i\rangle$  are i.i.d. uniformly random complex phases (and by convention,  $\langle \perp|U|\perp\rangle = 1$ ). Then, the algorithm's average linear XEB score on  $\sigma_R$  is identical to its average linear XEB score on  $|R\rangle$ , because of the invariance of the Haar measure with respect to phases.

## 13:8 The Quantum Supremacy Tsirelson Inequality

Next, we observe that one can prepare  $\sigma_R$  by measuring  $k$  copies of  $|\psi\rangle$  in the computational basis. We prove this in Lemma 12. So, when considering algorithms for XHOG that start with  $|R\rangle$ , it suffices to instead consider algorithms that simply measure  $k$  copies of  $|\psi\rangle$  in the computational basis. Such algorithms are much easier to analyze, because once we have measured the  $k$  copies of  $|\psi\rangle$ , we can assume (by convexity) that any optimal such algorithm for XHOG outputs a string  $z$  deterministically given the  $k$  measurement results. And in that case, clearly the optimal strategy is to output whichever  $z$  maximizes the posterior expectation of  $|\langle z|\psi\rangle|^2$  given the measurement results. We analyze this strategy in Lemma 13, and show that roughly  $2^{n/2}$  copies of  $|\psi\rangle$  are needed to solve  $b$ -XHOG for  $b$  bounded away from 2. The intuition is that the posterior expectation of  $|\langle z|\psi\rangle|^2$  increases only when we see  $z$  at least twice in the measurement results. However, the probability that any two measurement results are the same is tiny – on the order of  $2^{-n}$  – and so we need to measure at least  $2^{n/2}$  copies of  $|\psi\rangle$  to see any collisions with decent probability.

We now proceed to proving the necessary lemmas.

► **Lemma 12.** *Let  $|\psi\rangle = \sum_{i=1}^N \psi_i |i\rangle$  be an unknown quantum state, and consider a state  $|R\rangle$  of the form:*

$$|R\rangle := \bigotimes_{j=1}^k \alpha_j |\psi\rangle + \beta_j |\perp\rangle,$$

where  $\alpha_j, \beta_j$  are known for  $j \in [k]$ , and the vectors  $\{|1\rangle, |2\rangle, \dots, |N\rangle, |\perp\rangle\}$  form an orthonormal basis. Define the mixed state  $\sigma_R$  as above. Then there exists a protocol to prepare  $\sigma_R$  by measuring  $k$  copies of  $|\psi\rangle$  in the computational basis.

To give some intuition, we note that it is simpler to prove Lemma 12 in the case where  $\alpha_j = 1$  for all  $j$ . In that case,  $\sigma_R$  can be viewed as an  $N^k \times N^k$  density matrix where both the rows and columns are indexed by strings in  $[N]^k$ . Then, the averaging over diagonal unitaries implies that  $\sigma_R$  is obtained from  $(|R\rangle\langle R|)^{\otimes k}$  by zeroing out all entries where the index corresponding to the row is not a reordering of the index corresponding to the column. In fact, one can show that  $\sigma_R$  is expressible as a mixture of pure states, where each pure state is a uniform superposition over basis states that are reorderings of each other. Moreover, the probability associated with each pure state in this mixture is precisely the probability that one of the reorderings is observed when we measure  $k$  copies of  $|\psi\rangle$  in the computational basis. So, to prepare  $\sigma_R$ , it suffices to measure  $|\psi\rangle^{\otimes k}$  and then output the uniform superposition over reorderings of the measurement result.

The proof of Lemma 12 is similar, but we instead have to randomly set some of the measurement results to  $\perp$  with probability  $|\beta_j|^2$ .

Combining Lemma 11 and Lemma 12, we have reduced the problem of lower bounding the number of  $\mathcal{O}_\psi$  queries needed to solve  $b$ -XHOG, to lower bounding the number of copies of  $|\psi\rangle$  needed to solve  $b$ -XHOG. The next lemma lower bounds this latter quantity.

► **Lemma 13.** *Let  $|\psi\rangle$  be a Haar-random  $n$ -qubit quantum state. Consider a quantum algorithm that receives as input  $|\psi\rangle^{\otimes k}$  and outputs a string  $z \in \{0, 1\}^n$ . Then:*

$$\mathbb{E}_{|\psi\rangle, z} [|\langle z|\psi\rangle|^2] \leq \frac{2}{2^n} + \frac{O(k^2)}{4^n}.$$

We note that one should not expect Lemma 13 to be tight for large  $k$  (say,  $k = \Omega(2^{n/2})$ ). For example, to achieve  $b = 4$ , we need at least enough samples to see  $m \geq 3$  with good probability. But  $\Pr[m \geq 3]$  is negligible unless  $k = \Omega(2^{2n/3})$ . More generally, a tight bound

on the number of copies of  $|\psi\rangle$  needed to achieve a particular value of  $b$  seems closely related to the number of measurements of  $|\psi\rangle$  needed to see  $m \geq b - 1$ . This is like a sort of “balls into bins” problem [22, 27] with  $k$  balls and  $2^n$  bins, but where the probabilities associated to each bin follow a Dirichlet prior rather than being uniform.

We finally have the tools to prove the main result of this section.

► **Theorem 14.** *Any quantum query algorithm for  $(2 + \varepsilon)$ -XHOG with query access to  $\mathcal{O}_\psi$  for a Haar-random  $n$ -qubit state  $|\psi\rangle$  requires  $\Omega\left(\frac{2^{n/4} \varepsilon^{5/4}}{n}\right)$  queries.*

Lastly, we give an upper bound on the number of queries needed to nontrivially beat the naive algorithm for XHOG with  $\mathcal{O}_\psi$ . In fact, the following algorithm works with *any* oracle that prepares a Haar-random state (including a Haar-random unitary), because the algorithm only needs copies of  $|\psi\rangle$  and the ability to perform the reflection  $\mathcal{R}_\psi$ . We thank Scott Aaronson for suggesting this approach based on quantum collision-finding.

► **Theorem 15.** *There is a quantum algorithm for  $(2 + \Omega(1))$ -XHOG that makes  $O(2^{n/3})$  queries to a state preparation oracle for a Haar-random  $n$ -qubit state  $|\psi\rangle$ .*

## 4 Random State Preparation Oracles

In this section, we show that a canonical state preparation oracle and a random state preparation oracle are essentially equivalent, and use it to prove the quantum supremacy Tsirelson inequality for XHOG with a Haar-random oracle.

By Lemma 7, for a state  $|\psi\rangle$ , query access to a random state preparation oracle  $U_\psi$  implies query access to the canonical state preparation oracle  $\mathcal{O}_\psi$  with constant overhead. The reverse direction is less obvious. We know from the definition of  $U_\psi$  (Definition 8) that one can simulate  $U_\psi$  given *any*  $n$ -qubit unitary  $V$  that prepares  $|\psi\rangle$  from  $|0^n\rangle$ . So, it is tempting to let  $V = \mathcal{O}_\psi$  with  $|\perp\rangle = |0^n\rangle$  to argue that  $\mathcal{O}_\psi$  allows simulating  $U_\psi$ . However, this is only possible if  $|0^n\rangle$  is orthogonal to  $|\psi\rangle$ . And while we previously argued that we can always find a canonical state  $|\perp\rangle$  that is orthogonal to  $|\psi\rangle$  (Footnote 2), this requires extending the Hilbert space, so that  $\mathcal{O}_\psi$  no longer acts on  $n$  qubits!

To address this, imagine that we knew an explicit  $n$ -qubit state  $|\psi^\perp\rangle$  orthogonal to  $|\psi\rangle$ . Notice that we could perfectly swap  $|\psi\rangle$  and  $|\psi^\perp\rangle$ : the composition  $\mathcal{O}_\psi \mathcal{O}_{\psi^\perp} \mathcal{O}_\psi$  sends  $|\psi\rangle$  to  $|\psi^\perp\rangle$ ,  $|\psi^\perp\rangle$  to  $|\psi\rangle$ , and acts trivially on all states orthogonal to  $|\psi\rangle$  and  $|\psi^\perp\rangle$ . In particular, this swaps  $|\psi\rangle$  and  $|\psi^\perp\rangle$  while acting only on the space of  $n$ -qubit states. Next, if we know  $|\psi^\perp\rangle$  explicitly, we can certainly come up with an  $n$ -qubit unitary that sends  $|0^n\rangle$  to  $|\psi^\perp\rangle$ . By composing such a unitary with  $\mathcal{O}_\psi \mathcal{O}_{\psi^\perp} \mathcal{O}_\psi$ , we are left with an  $n$ -qubit unitary that sends  $|0^n\rangle$  to  $|\psi\rangle$ . This is sufficient to construct  $U_\psi$ , by Definition 8.

While we do not necessarily have such a state  $|\psi^\perp\rangle$ , a *random*  $n$ -qubit state  $|\varphi\rangle$  will be exponentially close to such a  $|\psi^\perp\rangle$  with overwhelming probability. The next theorem shows that we can use this observation to *approximately* simulate  $U_\psi$  given  $\mathcal{O}_\psi$ , by going through the steps above and keeping track of deviation from the ideal construction in terms of  $\langle\psi|\varphi\rangle$ .

► **Theorem 16.** *Let  $|\psi\rangle$  be an  $n$ -qubit state. Consider a quantum query algorithm  $A$  that makes  $T$  queries to  $U_\psi$ . Then there is a quantum query algorithm  $B$  that makes  $2T$  queries to  $\mathcal{O}_\psi$  such that:*

$$\left\| \mathbb{E}_{U_\psi} [A] - B \right\|_{\diamond} \leq \frac{10T + 4}{2^{n/2}}.$$

The above theorem implies that the oracle  $\mathcal{O}_\psi$  in Theorem 14 can be replaced by a Haar-random  $n$ -qubit unitary.

## 13:10 The Quantum Supremacy Tsirelson Inequality

► **Theorem 17.** *Any quantum query algorithm for  $(2 + \varepsilon)$ -XHOG with query access to  $U_\psi$  for a Haar-random  $n$ -qubit state  $|\psi\rangle$  (i.e. a Haar-random  $n$ -qubit unitary) requires  $\Omega\left(\frac{2^{n/4}\varepsilon^{5/4}}{n}\right)$  queries.*

### 5 Fourier Sampling Circuits

In this section, we prove the quantum supremacy Tsirelson inequality for single-query algorithms over FOURIER SAMPLING circuits.

Throughout this section, we let  $N = 2^n$ , and let  $\mathcal{F}_n := \{f : \{0, 1\}^n \rightarrow \{-1, 1\}\}$  denote the set of all  $n$ -bit Boolean functions. Given a function  $f \in \mathcal{F}_n$ , we define the Fourier coefficient

$$\hat{f}(z) := \frac{1}{2^n} \sum_{x \in \{0, 1\}^n} f(x)(-1)^{x \cdot z}$$

for each  $z \in \{0, 1\}^n$ . We also define the characters  $\chi_z : \{0, 1\}^n \rightarrow \{-1, 1\}$  for each  $z \in \{0, 1\}^n$ :

$$\chi_z(x) := (-1)^{x \cdot z}.$$

Given oracle access to a function  $f \in \mathcal{F}_n$ , the FOURIER SAMPLING quantum circuit for  $f$  consists of a layer of Hadamard gates, then a single query to  $f$ , then another layer of Hadamard gates, so that the resulting circuit samples a string  $z \in \{0, 1\}^n$  with probability  $\hat{f}(z)^2$ . In the context of XHOG, we consider the distribution of FOURIER SAMPLING circuits where the oracle  $f$  is chosen uniformly at random from  $\mathcal{F}_n$ .

► **Proposition 18.** *FOURIER SAMPLING circuits over  $n$  qubits solve  $(3 - \frac{2}{2^n})$ -XHOG.*

The following theorem shows the optimality of the 1-query algorithm for XHOG with FOURIER SAMPLING circuits:

► **Theorem 19.** *Any 1-query algorithm for  $b$ -XHOG over  $n$ -qubit FOURIER SAMPLING circuits satisfies  $b \leq 3 - \frac{2}{2^n}$ .*

To prove Theorem 19, we use the polynomial method of Beals et al. [10]. Consider a quantum query algorithm that makes  $T$  queries to  $f \in \mathcal{F}_n$  and outputs a string  $z \in \{0, 1\}^n$ . The polynomial method implies that for each  $z \in \{0, 1\}^n$ , the probability that the algorithm outputs  $z$  can be expressed as a real multilinear polynomial of degree  $2T$  in the bits of  $f$ . We write such a polynomial as:

$$p_z(f) = \sum_{S \subset \{0, 1\}^n, |S| \leq 2T} c_{z,S} \cdot \prod_{x \in S} f(x).$$

Then, the expected XEB score of this quantum query algorithm is given by:

$$\frac{1}{2^N} \sum_{f \in \mathcal{F}_n} \sum_{z \in \{0, 1\}^n} p_z(f) \cdot \hat{f}(z)^2. \quad (2)$$

Our key observation is that the quantity (2) is linear in the coefficients  $c_{z,S}$ . This allows us to express the largest XEB score achievable by polynomials of degree  $2T$  as a linear program, with the constraints that the polynomials  $\{p_z(f) : z \in \{0, 1\}^n\}$  must represent a probability distribution. Then, the objective value of the linear program can be upper bounded by giving a solution to the dual linear program.



## 6 Discussion

The most natural question left for future work is whether our bounds could be improved. Our lower bounds for  $b$ -XHOG with  $\mathcal{O}_\psi$  or  $U_\psi$  show that for constant  $\varepsilon$ ,  $(2 + \varepsilon)$ -XHOG requires  $\Omega\left(\frac{2^{n/4}}{\text{poly}(n)}\right)$  queries to either oracle, while the best upper bound we know of solves  $(2 + \varepsilon)$ -XHOG in  $O(2^{n/3})$  queries. We conjecture that this upper bound is tight.

One possible approach towards improving the lower bound for  $b$ -XHOG with  $\mathcal{O}_\psi$  (and by extension,  $U_\psi$ ) is to use the polynomial method, as we did for the FOURIER SAMPLING lower bound. Indeed, the output probabilities of an algorithm that makes  $T$  queries to  $\mathcal{O}_\psi$  can be expressed as degree- $2T$  polynomials in the entries of  $\mathcal{O}_\psi$ . If we write  $|\psi\rangle = \sum_{i=1}^N \alpha_i |i\rangle$ , then these are polynomials in the amplitudes  $\alpha_1, \dots, \alpha_N$  and the conjugates of the amplitudes  $\alpha_1^*, \dots, \alpha_N^*$ . Because of the invariance of the Haar measure with respect to phases, and because the linear XEB score depends only on the magnitudes of the amplitudes, we can further assume without loss of generality that the output probabilities are polynomials in the variables  $|\alpha_1|^2, \dots, |\alpha_N|^2$ , which are equivalently the measurement probabilities of  $|\psi\rangle$  in the computational basis. We can also assume that these polynomials are homogeneous, because the input variables satisfy  $\sum_{i=1}^N |\alpha_i|^2 = 1$ . Like in our FOURIER SAMPLING lower bound, the polynomials are constrained to represent a probability distribution for all valid inputs. However, unlike the FOURIER SAMPLING lower bound, this introduces uncountably many constraints in the primal linear program. It may still be possible to exhibit a solution to the dual linear program if only finitely many of the constraints are relevant (such an approach was used in [17], for example).

Our  $b$ -XHOG bound for FOURIER SAMPLING circuits is tight, but it only applies to single-query algorithms. In principle, our lower bound approach via the polynomial method could be generalized to algorithms that make additional queries, by increasing the degree of the polynomials in the linear program and exhibiting another dual solution. The challenge seems to be that the parity constraint on the monomials with nonzero coefficients becomes unwieldy when working with higher degree polynomials.

Beyond possible improvements to the query complexity bounds, it would be interesting to give some evidence that beating the naive XHOG algorithm is hard in the real world. Aaronson and Gunn [2] showed that  $(1 + \varepsilon)$ -XHOG is *classically* hard, assuming the classical hardness of nontrivially estimating the output probabilities of random quantum circuits. It is not clear whether a similar argument could work for quantum algorithms, though, because sampling from a random quantum circuit gives a better-than-trivial algorithm for estimating its output probabilities.

---

## References

- 1 Scott Aaronson and Lijie Chen. Complexity-theoretic foundations of quantum supremacy experiments. In *Proceedings of the 32nd Computational Complexity Conference, CCC '17*, Dagstuhl, DEU, 2017. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.
- 2 Scott Aaronson and Sam Gunn. On the classical hardness of spoofing linear cross-entropy benchmarking. *Theory of Computing*, 16(11):1–8, 2020. doi:10.4086/toc.2020.v016a011.
- 3 Scott Aaronson, Robin Kothari, William Kretschmer, and Justin Thaler. Quantum Lower Bounds for Approximate Counting via Laurent Polynomials. In Shubhangi Saraf, editor, *35th Computational Complexity Conference (CCC 2020)*, volume 169 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 7:1–7:47, Dagstuhl, Germany, 2020. Schloss Dagstuhl–Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.CCC.2020.7.

- 4 Dorit Aharonov, Alexei Kitaev, and Noam Nisan. Quantum circuits with mixed states. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, STOC '98, page 20–30, New York, NY, USA, 1998. Association for Computing Machinery. doi:10.1145/276698.276708.
- 5 Andris Ambainis. Understanding quantum algorithms via query complexity. In *Proceedings of the 2018 International Congress of Mathematicians*, volume 3, pages 3249–3270, 2018.
- 6 Andris Ambainis, Loïck Magnin, Martin Roetteler, and Jérémie Roland. Symmetry-assisted adversaries for quantum state generation. In *2011 IEEE 26th Annual Conference on Computational Complexity*, pages 167–177, 2011.
- 7 Andris Ambainis, Ansis Rosmanis, and Dominique Unruh. Quantum attacks on classical proof systems: The hardness of quantum rewinding. In *Proceedings of the 2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, FOCS '14, page 474–483, USA, 2014. IEEE Computer Society. doi:10.1109/FOCS.2014.57.
- 8 Srinivasan Arunachalam, Aleksandrs Belovs, Andrew M. Childs, Robin Kothari, Ansis Rosmanis, and Ronald de Wolf. Quantum Coupon Collector. In Steven T. Flammia, editor, *15th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2020)*, volume 158 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 10:1–10:17, Dagstuhl, Germany, 2020. Schloss Dagstuhl–Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.TQC.2020.10.
- 9 Frank Arute, Kunal Arya, Ryan Babbush, et al. Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779):505–510, 2019. doi:10.1038/s41586-019-1666-5.
- 10 Robert Beals, Harry Buhrman, Richard Cleve, Michele Mosca, and Ronald de Wolf. Quantum lower bounds by polynomials. *J. ACM*, 48(4):778–797, July 2001. doi:10.1145/502090.502097.
- 11 John Bell. On the Einstein-Podolsky-Rosen paradox. *Physics*, 1:195–200, November 1964. doi:10.1103/PhysicsPhysiqueFizika.1.195.
- 12 Aleksandrs Belovs. Variations on quantum adversary, 2015. arXiv:1504.06943.
- 13 Aleksandrs Belovs and Ansis Rosmanis. Approximate unitary t-designs by short random quantum circuits using nearest-neighbor and long-range gates, 2020. arXiv:2002.06879.
- 14 Fernando G. S. L. Brandão, Aram W. Harrow, and Michał Horodecki. Local random quantum circuits are approximate polynomial-designs. *Communications in Mathematical Physics*, 346(2):397–434, 2016. doi:10.1007/s00220-016-2706-8.
- 15 Gilles Brassard, Peter Høyer, Michele Mosca, and Alain Tapp. Quantum amplitude amplification and estimation. In *Quantum Computation and Quantum Information*, volume 305 of *Contemporary Mathematics*, pages 53–74. American Mathematical Society, 2002.
- 16 Gilles Brassard, Peter Høyer, and Alain Tapp. Quantum cryptanalysis of hash and claw-free functions. *SIGACT News*, 28(2):14–19, June 1997. doi:10.1145/261342.261346.
- 17 Mark Bun and Justin Thaler. Dual lower bounds for approximate degree and markov-bernstein inequalities. *Inf. Comput.*, 243(C):2–25, August 2015. doi:10.1016/j.ic.2014.12.003.
- 18 Boris Cirel'son (Tsirelson). Quantum generalizations of Bell's inequality. *Letters in Mathematical Physics*, 4(2):93–100, 1980. doi:10.1007/BF00417500.
- 19 John F. Clauser, Michael A. Horne, Abner Shimony, and Richard A. Holt. Proposed experiment to test local hidden-variable theories. *Phys. Rev. Lett.*, 23:880–884, October 1969. doi:10.1103/PhysRevLett.23.880.
- 20 Richard Cleve, Peter Høyer, Benjamin Toner, and John Watrous. Consequences and limits of nonlocal strategies. In *Proceedings of the 19th IEEE Annual Conference on Computational Complexity*, CCC '04, page 236–249, USA, 2004. IEEE Computer Society.
- 21 Aram Harrow and Saeed Mehraban. Approximate unitary t-designs by short random quantum circuits using nearest-neighbor and long-range gates, 2018. arXiv:1809.06957.
- 22 Norman L. Johnson and Samuel Kotz. *Urn models and their application: an approach to modern discrete probability theory*. Wiley, 1977.

- 23 Shelby Kimmel, Cedric Yen-Yu Lin, Guang Hao Low, Maris Ozols, and Theodore J. Yoder. Hamiltonian simulation with optimal sample complexity. *npj Quantum Information*, 3(1):13, 2017. doi:10.1038/s41534-017-0013-7.
- 24 Troy Lee, Rajat Mittal, Ben W. Reichardt, Robert Špalek, and Mario Szegedy. Quantum query complexity of state conversion. In *Proceedings of the 2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, FOCS '11, page 344–353, USA, 2011. IEEE Computer Society. doi:10.1109/FOCS.2011.75.
- 25 Nathan Lindzey and Ansis Rosmanis. A Tight Lower Bound For Non-Coherent Index Erasure. In Thomas Vidick, editor, *11th Innovations in Theoretical Computer Science Conference (ITCS 2020)*, volume 151 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 59:1–59:37, Dagstuhl, Germany, 2020. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.ITCS.2020.59.
- 26 Frederic Magniez, Ashwin Nayak, Jeremie Roland, and Miklos Santha. Search via quantum walk. In *Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing*, STOC '07, page 575–584, New York, NY, USA, 2007. Association for Computing Machinery. doi:10.1145/1250790.1250874.
- 27 Martin Raab and Angelika Steger. “Balls into bins” - a simple and tight analysis. In *Proceedings of the Second International Workshop on Randomization and Approximation Techniques in Computer Science*, RANDOM '98, pages 159–170, Berlin, Heidelberg, 1998. Springer-Verlag.
- 28 Ben W. Reichardt. Reflections for quantum query algorithms. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '11, page 560–569, USA, 2011. Society for Industrial and Applied Mathematics.



# Approximately Strategyproof Tournament Rules in the Probabilistic Setting

Kimberly Ding

Computer Science, Princeton University, NJ, USA  
kding@princeton.edu

S. Matthew Weinberg

Computer Science, Princeton University, NJ, USA  
smweinberg@princeton.edu

---

## Abstract

We consider the manipulability of tournament rules which map the results of  $\binom{n}{2}$  pairwise matches and select a winner. Prior work designs simple tournament rules such that no pair of teams can manipulate the outcome of their match to improve their probability of winning by more than  $1/3$ , and this is the best possible among any Condorcet-consistent tournament rule (which selects an undefeated team whenever one exists) [14, 15]. These lower bounds require the manipulators to know precisely the outcome of all future matches.

We take a beyond worst-case view and instead consider tournaments which are “close to uniform”: the outcome of all matches are independent, and no team is believed to win any match with probability exceeding  $1/2 + \varepsilon$ . We show that Randomized Single Elimination Bracket [14] and a new tournament rule we term Randomized Death Match have the property that no pair of teams can manipulate the outcome of their match to improve their probability of winning by more than  $\varepsilon/3 + 2\varepsilon^2/3$ , for all  $\varepsilon$ , and this is the best possible among any Condorcet-consistent tournament rule.

Our main technical contribution is a recursive framework to analyze the manipulability of certain forms of tournament rules. In addition to our main results, this view helps streamline previous analysis of Randomized Single Elimination Bracket, and may be of independent interest.

**2012 ACM Subject Classification** Theory of computation → Algorithmic game theory

**Keywords and phrases** Tournaments, Incentive Compatibility, Recursive Analysis, Social Choice Theory

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.14

**Funding** S. Matthew Weinberg: Supported by NSF CAREER Award CCF-1942497.

## 1 Introduction

A tournament consists of  $n$  teams competing to win a championship via pairwise matches, and a tournament rule (possibly randomly) selects a single winner as a result of these matches. Tournament rules have been studied within Social Choice Theory for decades [6, 12, 16, 3, 7, 10, 11], see also [4] for a survey, and have gained attention from a few angles within TCS more recently [2, 1, 18, 17, 9, 8, 14, 15]. Our work follows the model studied in [2, 1, 14, 15] and seeks to design tournaments which are both fair (in that they select a reasonable winner, based on the match outcomes), and “as strategyproof as possible” subject to this.

More specifically, these works acknowledge that an undefeated team, if one exists, should surely win any reasonable tournament format. Formally, this property is termed *Condorcet-consistent* (Definition 4). These works also consider the possibility of two teams strategically manipulating the match between them to improve the probability that one of them wins. The situation to have in mind is that perhaps two teams sponsored by the same company enter an eSports tournament, and wish to maximize the probability that either of them take home the prize money. A rule is 2-Strongly Non-Manipulable (2-SNM, Definition 7) if no pair of teams can manipulate it to improve the probability that one of them wins.



© Kimberly Ding and S. Matthew Weinberg;  
licensed under Creative Commons License CC-BY  
12th Innovations in Theoretical Computer Science Conference (ITCS 2021).  
Editor: James R. Lee; Article No. 14; pp. 14:1–14:20



Leibniz International Proceedings in Informatics  
LIPIcs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Initial works quickly established that no tournament rule exists which is both Condorcet-consistent and 2-SNM [2, 1]. More recent works study the extent to which tournament rules can be Condorcet-consistent and *approximately* 2-SNM. Specifically, a rule is 2-SNM- $\alpha$  if no pair of teams can improve the probability that one of them wins by more than  $\alpha$  (Definition 7). [14, 15] design simple Condorcet-consistent tournament rules (defined in Section 3) which are 2-SNM-1/3, and also show this is the best possible *worst-case* guarantee.

### 1.1 Our Results: Probabilistic Tournaments

The lower bounds in previous works assume that the *deterministic future outcome* of all matches is known at the time of manipulation. While this is certainly a plausible scenario, most competitions worth watching have some element of uncertainty, even for matches between strong and weak teams. Indeed, [14, Open Problem 2] explicitly asks whether improved guarantees are possible if instead the teams have a common Bayesian prior about the possible outcomes of future matches which is *bounded away from deterministic*.

For example, consider the case where the outcome of all matches are uniformly at random. Then it is not hard to design a Condorcet-consistent tournament rule which is 2-SNM in this case. One example is a simple single-elimination bracket: when two manipulating teams face each other, each of them is equally likely to continue on and win the tournament, so manipulating doesn't help.

What if instead the outcome of all matches are not uniformly random, but close? More specifically, what if the match results are independent, and no team wins any match with probability more than  $1/2 + \varepsilon$ ? When  $\varepsilon = 0$ , the previous paragraph establishes that rules exist where profitable manipulation is impossible. When  $\varepsilon = 1/2$ , [14, 15] establish that 2-SNM-1/3 tournaments exist, but no better. What about when  $\varepsilon \in (0, 1/2)$ ? How do the achievable guarantees vary as a function of  $\varepsilon$ ?

Our main result resolves this question, and nails down the guarantees precisely as a function of  $\varepsilon$ . Moreover, we show that *the same tournament rule achieves the optimal guarantee for all  $\varepsilon$* . Below, Randomized Single Elimination Bracket (Definition 12, henceforth RSEB) randomly seeds all teams, then runs a single-elimination bracket to determine the winner, and was shown to be 2-SNM-1/3 in [14]. Randomized Death Match (Definition 14, henceforth RDM) repeatedly picks two uniformly random teams to play a match, and eliminates the loser (and is first analyzed in this paper).

► **Informal Theorem 1** (See Theorems 30, 31). *For all  $\varepsilon \in [0, 1/2]$ , Randomized Death Match and Randomized Single Elimination Bracket are 2-SNM- $(\varepsilon/3 + 2\varepsilon^2/3)$  when match outcomes are independent, and no team wins any match with probability more than  $1/2 + \varepsilon$ . Moreover, for all  $\varepsilon \in [0, 1/2]$ , no Condorcet-consistent tournament rule is 2-SNM- $\alpha$  for any  $\alpha < \varepsilon/3 + 2\varepsilon^2/3$ , on the set of all tournaments with independent match outcomes that no team wins with probability more than  $1/2 + \varepsilon$ .*

### 1.2 Technical Highlights

We prove our main result by finding a strong structural similarity between tournaments like RDM and RSEB: they can be defined recursively. Specifically, RDM could be alternatively defined as “Pick two teams uniformly at random, and eliminate the loser of their match. Then, recurse on the remaining teams.” Similarly, RSEB can be alternatively defined as “Pick a uniformly random perfect matching between the teams, and eliminate all teams which

lose their match. Then, recurse on the remaining teams.” Even the Randomized King of the Hill rule (Definition 13, henceforth RKotH) defined by [15] fits this framework as well: “Pick a uniformly random team to play all other teams. Eliminate all teams who lose a match, and recurse on the remaining teams.”

In Section 3, we give a formal definition of what it means to be a recursive tournament rule. And in Section 4 (specifically, Theorem 21), we provide a general framework to analyze the manipulability of recursive tournament rules on probabilistic tournaments. This provides a fairly clean outline to analyze recursive tournament rules, and our main result then applies this framework to RDM and RSEB. In the  $\varepsilon = 1/2$  case, our analysis of RSEB in isolation is perhaps not much simpler than that of [14], but our proof is arguably more structured. Indeed, a substantial fraction of our proof can be applied verbatim to other tournament rules like RDM, or applied verbatim to the  $\varepsilon < 1/2$  case.

It is worth noting that our framework does face some technical barriers in accommodating RKotH (and we leave open whether RKotH achieves the same guarantees as RDM and RSEB). But, the technical barrier is easy to describe: the matches played in each round of RDM and RSEB form a matching – no team plays more than one match. In RKotH, some team plays multiple matches. It seems likely that our analysis would extend (perhaps with messier calculations) to any recursive rule where each round’s matches form a matching. But we highlight the aspects of our analysis which rely on this aspect of RDM/RSEB (and therefore don’t hold for RKotH), and believe this is a genuine barrier.

### 1.3 Further Related Work

We’ve already discussed the most related work [2, 1, 14, 15]. The model is first posed in [2], and [1] design tournaments which are 2-SNM and approximately Condorcet-consistent (e.g. pick a uniformly random match and declare the winner of that match to win the tournament). [14] first proposed to instead consider rules which are Condorcet-consistent and approximately strategyproof, and establishes that RSEB is 2-SNM-1/3 and that this is optimal. [15] considers larger manipulating sets (not relevant to this paper) and also designs RKotH, showing that it too is 2-SNM-1/3 and satisfies a stronger notion of fairness termed “cover-consistent”. In relation to these works, our main contribution is going beyond the worst-case to derive improved bounds when match outcomes are more uncertain. A technical contribution is our framework of recursive tournament rules.

Other recent works within TCS focus specifically on single-elimination brackets and manipulation in the form of a bracket designer trying to get a certain team to win [18, 17, 9, 8], or manipulability of particular tournament rules such as the World Cup qualifying procedure [13, 5]. Aside from being thematically related, there is no significant technical overlap with our work.

### 1.4 Roadmap

Section 2 immediately follows with definitions and preliminaries. Section 3 provides definitions concerning recursive tournament rules, and formally defines RDM and RSEB. Section 4 provides our framework for analyzing the manipulability of recursive tournament rules. Section 5 applies this framework, as a warmup, to rederive the main result of [14] and analyze RDM/RSEB in the deterministic case. Section 6 proves our main results, and Section 7 provides a brief conclusion.



## 2 Preliminaries

### 2.1 Tournament Rule Basics

In this section, we introduce notation consistent with prior work [1, 14, 15].

► **Definition 2** (Deterministic Tournament). *A (round robin) tournament  $T$  on  $n$  teams is a complete, directed graph on  $n$  vertices whose edges denote the outcome of a match between two teams. Team  $i$  beats team  $j$  if the edge between them points from  $i$  to  $j$ .*

► **Definition 3** (Tournament Rule). *A tournament rule  $r$  is a function that maps (deterministic) tournaments  $T$  to a distribution over teams, where  $r_i(T) := \Pr[r(T) = i]$  denotes the probability that team  $i$  is declared the winner of tournament  $T$  under rule  $r$ . We use the shorthand  $r_S(T) := \sum_{i \in S} r_i(T)$  to denote the probability that a team in  $S$  is declared the winner of tournament  $T$  under rule  $r$ .*

Finally, we are interested in tournament rules which satisfy basic notions of fairness. Importantly, note that Condorcet-consistence is a minimal notion of fairness, and in particular does not constrain the behavior of  $r$  on any tournament without a Condorcet winner.

► **Definition 4** (Condorcet-Consistent). *Team  $i$  is a Condorcet winner of a tournament  $T$  if  $i$  beats every other team (under  $T$ ). A tournament rule  $r$  is Condorcet-consistent if for every tournament  $T$  with a Condorcet winner  $i$ ,  $r_i(T) = 1$  (whenever  $T$  has a Condorcet winner, that team wins with probability 1).*

### 2.2 Independent Probabilistic Tournaments

In this work, we study probabilistic tournaments. That is, we are interested in tournaments where the outcome of each match is not known to teams “in advance”, but teams share a Bayesian prior about the likelihood of each possible outcome. In particular, we consider when match outcomes are independent.

► **Definition 5** (Independent Probabilistic Tournament). *A probabilistic tournament  $T$  is just a distribution over deterministic tournaments. For notational convenience, we slightly abuse notation and refer by  $r_i(T)$  to  $\mathbb{E}[r_i(T)]$  (that is,  $r_i(T)$  is the probability that team  $i$  wins when rule  $r$  is applied to  $T$ , over the randomness in  $r$  and the randomness in drawing  $T$ ). A probabilistic tournament  $T$  is independent if all match outcomes in  $T$  are independent events. Observe that a probabilistic tournament  $T$  is fully defined by probabilities  $p_{ij}^T$  for all  $i < j$ , where  $p_{ij}^T$  denotes the probability that  $i$  beats  $j$  in tournament  $T$ .*

Observe that deterministic tournaments are also independent probabilistic tournaments, with each  $p_{ij}^T \in \{0, 1\}$ . Like prior work, we study tournament rules which are “as strategyproof as possible”. Because of our focus on independent probabilistic tournaments, we first refine previous definitions of non-manipulability.

► **Definition 6** ( $S$ -adjacent). *Two independent probabilistic tournaments  $T, T'$  are  $S$ -adjacent if  $p_{ij}^T = p_{ij}^{T'}$  whenever  $\{i, j\} \not\subseteq S$ . That is, two independent probabilistic tournaments are  $S$ -adjacent when all (probabilistic) match outcomes are identical, except possibly for matches between two teams in  $S$ .*

Intuitively, two tournaments  $T, T'$  are  $S$ -adjacent if the set of teams  $S$  can manipulate the outcomes of matches between them in advance and cause the resulting (probabilistic) tournament to go from  $T$  to  $T'$ .

► **Definition 7** (Manipulating a Tournament). *For a set  $S$  of teams, independent probabilistic tournament  $T$ , and tournament rule  $r$ , we define  $\alpha_S^r(T)$  to be the maximum winning probability that  $S$  can possibly gain by manipulating  $T$  to an  $S$ -adjacent  $T'$ . That is:  $\alpha_S^r(T) := \max_{T': T' \text{ is } S\text{-adjacent to } T} \{r_S(T') - r_S(T)\}$ .*

*For a class  $\mathcal{T}$  of (independent probabilistic) tournaments, we also define  $\alpha_k^r(\mathcal{T}) := \max_{T \in \mathcal{T}, S: |S| \leq k} \{\alpha_S^r(T)\}$ . If  $\alpha_k^r(\mathcal{T}) \leq \alpha$ , we say  $r$  is  $k$ -Strongly Non-Manipulable at probability  $\alpha$  with respect to  $\mathcal{T}$  ( $k$ -SNM $_{\mathcal{T}-\alpha}$ ). To match notation of prior work, we also say a tournament is  $k$ -SNM- $\alpha$  if it is  $k$ -SNM $_{S^{1/2}-\alpha}$ .<sup>1</sup>*

*Finally, we also define  $\alpha_k(\mathcal{T}) = \inf_{\text{Condorcet consistent } r} \{\alpha_k^r(\mathcal{T})\}$ .*

Intuitively,  $r$  is  $k$ -SNM $_{\mathcal{T}-\alpha}$  if no colluding set of  $\leq k$  teams can manipulate a tournament in  $\mathcal{T}$  to improve the probability the winner is in  $S$  by more than  $\alpha$ . The refinement over prior work is that the condition only holds for tournaments in  $\mathcal{T}$  – prior work only considers guarantees that hold over *all* tournaments. The additional notation in Definition 7 are just terms that will be helpful for later exposition.

We focus on independent probabilistic tournaments that are close to uniformly random.

► **Definition 8** ( $\varepsilon$ -Bounded Tournaments). *An independent probabilistic tournament  $T$  is weakly  $\varepsilon$ -bounded if for all  $i, j$ ,  $p_{ij}^T \in [1/2 - \varepsilon, 1/2 + \varepsilon]$ . We refer to  $\mathcal{T}^\varepsilon$  as the set of all  $\varepsilon$ -bounded tournaments.*

*An independent probabilistic tournament  $T$  is strictly  $\varepsilon$ -bounded if for all  $i, j$ ,  $p_{ij}^T \in \{1/2 - \varepsilon, 1/2 + \varepsilon\}$ . We refer to  $\mathcal{S}^\varepsilon$  as the set of all strictly  $\varepsilon$ -bounded tournaments. It will be helpful to define the notation  $\mathcal{T}_{\leq n}^\varepsilon$  (respectively,  $\mathcal{S}_{\leq n}^\varepsilon$ ) as the set of all weakly- (respectively, strictly-) bounded tournaments on  $\leq n$  teams.*

For example, every deterministic tournament is  $1/2$ -bounded, and the uniformly random tournament is  $0$ -bounded. Our main results study  $\alpha_k(\mathcal{T}^\varepsilon)$  as a function of  $\varepsilon$ . We conclude with a brief lemma relating  $\alpha_k(\mathcal{T}^\varepsilon)$  to  $\alpha_k(\mathcal{S}^\varepsilon)$ , as direct analysis of  $\alpha_k(\mathcal{S}^\varepsilon)$  is significantly simpler than direct analysis of  $\alpha_k(\mathcal{T}^\varepsilon)$ .

► **Proposition 9.** *For all rules  $r$ , and all  $\varepsilon, k$ ,  $\alpha_k^r(\mathcal{T}^\varepsilon) = \alpha_k^r(\mathcal{S}^\varepsilon)$ . Therefore, for all  $\varepsilon, k$ ,  $\alpha_k(\mathcal{T}^\varepsilon) = \alpha_k(\mathcal{S}^\varepsilon)$ .*

While the proof is deferred to Appendix A, the high-level outline is fairly intuitive. First, the “Therefore, . . .” statement follows trivially from the first portion of the lemma. Also, it is trivial to see that  $\alpha_k^r(\mathcal{T}^\varepsilon) \geq \alpha_k^r(\mathcal{S}^\varepsilon)$ , as  $\mathcal{S}^\varepsilon \subseteq \mathcal{T}^\varepsilon$ . So the interesting step is establishing  $\alpha_k^r(\mathcal{T}^\varepsilon) \leq \alpha_k^r(\mathcal{S}^\varepsilon)$ . Intuitively, this follows because all (independent probabilistic) tournaments in  $\mathcal{T}^\varepsilon$  can be written as convex combinations of tournaments in  $\mathcal{S}^\varepsilon$ , and one might expect that any particular tournament rule is most manipulable on extreme points (indeed this is true). With Proposition 9, we may restrict our study to  $\alpha_k(\mathcal{S}^\varepsilon)$ .

### 3 Recursive Tournament Rules

In this section, we formalize a class of tournament rules which have a recursive aspect to them. This will help us streamline previous analysis of [14], and also easily design a new tournament rule with matching guarantees. In addition, this view will help give us a clean outline to analyze the performance of these rules on tournaments in  $\mathcal{T}^\varepsilon$ , rather than just in the worst case.

<sup>1</sup> Note that  $\mathcal{S}^{1/2}$  is the set of all deterministic tournaments, defined shortly below.

► **Definition 10** (Elimination Rule). *An elimination rule  $E$  takes as input a number  $n$  of teams and selects (possibly randomly) a set  $M := E(n)$  of matches to play, with  $|M| \in [1, n - 1]$ .*

*An elimination rule is matching if  $M$  is a (not necessarily perfect) matching with probability one.*

► **Definition 11** (Recursive Tournament Rule). *A recursive tournament rule is fully defined by its elimination rule  $E$ . The recursive tournament rule  $r^E$  takes as input a tournament  $T$ , samples matches  $M := E(n)$  to play, and then eliminates any team which loses a match in  $M$ .<sup>2</sup> Specifically,  $T|_M$  denotes the induced subgraph of  $T$  on teams not eliminated by the matches in  $M$ . The tournament then recursively executes  $r^E(T|_M)$  to select a winner. As a base case, when there is only one team left, that team is the winner.*

We now give three examples of elimination rules, and the resulting tournament rule. Randomized Single Elimination Bracket was first studied in [14], Randomized King of the Hill was first studied in [15], and Randomized Death Match is first studied in this paper.

► **Definition 12** (Randomized Single Elimination Bracket). *For a tournament  $T$  on  $n$  teams, let  $n' := 2^{\lceil \log_2 n \rceil}$ . Create  $n' - n$  dummy players who all lose to the original  $n$  teams. Let  $M$  be matches corresponding to a uniformly random perfect matching (i.e. exactly  $n'/2$  matches are played, and every team plays in exactly one match). Eliminate the losers and recurse on the remaining (non-dummy) teams.*

► **Definition 13** (Randomized King of the Hill). *Pick a uniformly random team  $i$ , to play all others. Observe that if  $i$  is a Condorcet winner, then  $i$  will be the only remaining team. Otherwise,  $i$  and every team it beats will be eliminated. Recurse on the remaining teams.*

► **Definition 14** (Randomized Death Match). *Pick two uniformly random teams (without replacement) and play their match. Eliminate the loser and recurse on the remaining teams.*

Observe that our definition of Randomized Single Elimination Bracket (RSEB) differs semantically from that given in [14], where the  $n'$  teams are uniformly permuted into  $n'$  seeds, and then the resulting bracket is played (without re-randomizing at each round). Observe that the two definitions are equivalent (identically distributed), however, as our definition simply produces the seeding by first figuring out the first-round matches, then the second-round matches, etc. Our definition of Randomized King of the Hill (RKotH) is identical (semantically) to that given in [15].

Randomized Death Match (RDM) is similar to Randomized Voting Caterpillar (RVC) [14]. Like RDM, RVC picks two uniformly random teams (without replacement) and eliminates the loser. However, rather than a “pure recursion”, RVC proceeds by picking *one* uniformly random remaining team to play the previous winner. This subtle distinction causes RDM to be 2-SNM-1/3 (Theorem 23), but not RVC ([14, Theorem 3.14]). Observe also that RSEB and RDM have matching elimination rules, but RKotH doesn’t (this is the main technical barrier in extending our analysis to RKotH).

Finally, observe that all three rules above are *anonymous*: relabeling the teams simply relabels the distribution over winners. This will play a role in our analysis.

► **Definition 15** (Anonymous). *A tournament rule  $r$  is anonymous if for every tournament  $T$ , and every permutation  $\sigma$ , and all  $i$ ,  $r_{\sigma(i)}(\sigma(T)) = r_i(T)$ .*

<sup>2</sup> Observe that because  $|M| \in [1, n - 1]$  that at least one team is eliminated, but not all teams are eliminated.

## 4 Key Framework

In this section, we propose an outline to analyze the manipulability of recursive tournament rules for independent probabilistic tournaments. For notational convenience, if a team  $i$  is not present in tournament  $T$  (e.g. because they were eliminated in an earlier round), we abuse notation and denote by  $r_i(T) := 0$ . Additionally, we let  $M(T)$  denote the outcome of matches  $M$  for tournament  $T$  (i.e. for all  $(u, v) \in M$ , whether  $u$  or  $v$  wins in tournament  $T$ ). Note that if  $T$  is probabilistic,  $M(T)$  is a random variable, even after conditioning on  $M$ .

Our first step simply observes that recursive tournament rules can be analyzed recursively due to linearity of expectation. A (short) proof of Lemma 16 appears in Appendix B.

► **Lemma 16.** *Let  $r^E$  be any recursive tournament rule. Let  $T, T'$  be independent probabilistic tournaments on  $n > 1$  teams, and let  $S$  be any subset of teams. Then:*

$$r_S^E(T') - r_S^E(T) = \mathbb{E}_{M \leftarrow E(n)}[r_S^E(T'|_M) - r_S^E(T|_M)].$$

To help parse notation: recall that if  $T, T'$  are not deterministic, then the notation  $r_S^E(T')$  (respectively,  $r_S^E(T), r_S^E(T'|_M), r_S^E(T|_M)$ ) is taking an expectation over  $T'$  (respectively,  $T, T'|_M, T|_M$ ), as per Definition 5.<sup>3</sup> On the right-hand side, we are taking an expectation first over the matches  $M$  which are played. Inside the expectation, the teams in  $T|_M$  and  $T'|_M$  are still random variables (because they depend on the *outcome* of matches in  $M$  in  $T$ , which are still random after conditioning on  $M$ ). And after the teams are determined by  $M(T), M(T')$ , the tournaments  $T'|_M, T|_M$  are still probabilistic tournaments.

Importantly, observe that when  $S = \{u, v\}$  and  $T, T'$  are  $S$ -adjacent, the tournaments  $T|_M$  and  $T'|_M$  may differ for two reasons. First, perhaps  $(u, v) \in M$ . In this case, perhaps  $M(T) \neq M(T')$  (because  $T$  and  $T'$  can differ on the  $(u, v)$  match), and then  $T|_M, T'|_M$  may have different sets of teams. Second, perhaps  $(u, v) \notin M$ , implying that  $M(T) = M(T')$  (because  $T$  and  $T'$  are identically distributed outside of the  $(u, v)$  match).  $T|_M$  and  $T'|_M$  therefore have the same sets of teams, but it can still be that  $p_{uv}^T \neq p_{uv}^{T'}$ , so the tournaments  $T|_M$  and  $T'|_M$  can still differ due to this match (if both  $u$  and  $v$  are not eliminated).

The second step in our framework simply splits the recursive analysis into cases based on  $M$ , and the results of the matches  $M(T)$ . We define these cases clearly below, and then state our main framework.

► **Definition 17 (Base Case).** *We say that tournament  $T$  is a base case if  $\alpha_S^r(T) = 0$ . That is, it is not possible for  $S$  to gain by manipulating tournament  $T$  under rule  $r$ .*

One clear base case occurs if  $S = \{u, v\}$ , but  $u$  is not even in  $T$ . Lemma 22 later identifies another for anonymous tournament rules (if relabeling  $u$  and  $v$  doesn't change  $T$  except for the  $(u, v)$  match, then  $\{u, v\}$  cannot gain by manipulating an anonymous tournament rule).

► **Definition 18 (Bad Terminal Event).**  *$M$  is a bad terminal event if  $(u, v) \in M$ . That is, a bad terminal event occurs when the  $(u, v)$  match is played this round. We denote the occurrence of this event by  $B$ .*

► **Definition 19 (Good Terminal Event).**  *$M, M(T)$  is a good terminal event if  $(u, v) \notin M$ , and  $M(T)$  is such that  $T|_M$  is a base case.<sup>4</sup> That is, the  $(u, v)$  match is not played now (so no gains from manipulation are possible), and no future gains are possible either. We denote the occurrence of this event by  $G$ .*

<sup>3</sup> In particular, note that  $T|_M$  and  $T'|_M$  are not independent probabilistic tournaments, but are still probabilistic tournaments, as they are distributions over independent probabilistic tournaments.

<sup>4</sup> Observe that because  $(u, v) \notin M$ , that if  $T'$  is  $S$ -adjacent to  $T$  then  $M(T)$  and  $M(T')$  are identically distributed.

► **Definition 20** (Recursive Event).  $M, M(T)$  is a recursive event if  $(u, v) \notin M$ , but  $M(T)$  is not such that  $T|_M$  is a base case. That is, a recursive event occurs when the  $(u, v)$  match is not played now, but may be played later (and manipulating it may be beneficial). We denote the occurrence of this event by  $R$ .

We now state the theorem which drives our analysis. Observe that the bound claimed by Theorem 21 could in principle be (very) loose, but our subsequent sections show that this framework suffices to nail down  $\alpha_2^r(\mathcal{T}^\varepsilon)$  for RDM and RSEB, and also that these are the best possible guarantees for any Condorcet-consistent tournament rule.

► **Theorem 21.** Let  $r^E$  be a recursive tournament rule,  $\varepsilon \in [0, 1/2]$ , and  $S := \{u, v\}$ . Let  $b, g, c \geq 0$  (with  $b + g > 0$ ) be so that for any  $T \in \mathcal{S}^\varepsilon$ , and any  $T'$  which is  $S$ -adjacent to  $T$ :

- $\Pr_{M \leftarrow E(n)}[B] \leq b.$
- $\Pr_{M \leftarrow E(n), M(T)}[G] \geq g.$
- $\mathbb{E}_{M \leftarrow E(n)}[r_S^E(T'|_M) - r_S^E(T|_M)|B] \leq c.$

Then:  $\alpha_2^{r^E}(\mathcal{T}^\varepsilon) \leq \frac{bc}{b+g}.$

**Proof.** We prove the theorem by induction, and focus on  $\mathcal{S}^\varepsilon$  first (extending to  $\mathcal{T}^\varepsilon$  using Proposition 9). As a base case, observe that when there are at most two teams remaining, no gains from manipulation are possible and therefore  $\alpha_2^{r^E}(\mathcal{S}_{\leq 2}^\varepsilon) \leq 0 \leq \frac{bc}{b+g}$  as desired.

For the inductive hypothesis, assume that  $\alpha_2^{r^E}(\mathcal{S}_{\leq n-1}^\varepsilon) \leq \frac{bc}{b+g}$  and consider now any tournament  $T \in \mathcal{S}^\varepsilon$  on  $n$  teams, and an  $S$ -adjacent  $T'$ . We have the following chain of equalities, which essentially just breaks down  $r_S^E(T') - r_S^E(T)$  based on the three events  $R, B, G$  (below,  $I(X)$  denotes the indicator random variable for event  $X$ , which is 1 when event  $X$  occurs and 0 otherwise):

$$\begin{aligned}
 r_S^E(T') - r_S^E(T) &= \mathbb{E}_{M \leftarrow E(n)}[r_S^E(T'|_M) - r_S^E(T|_M)] \\
 &= \mathbb{E}_{M \leftarrow E(n), M(T)}[(r_S^E(T'|_M) - r_S^E(T|_M)) \cdot (I(R) + I(B) + I(G))] \\
 &= \Pr_{M \leftarrow E(n), M(T)}[R] \cdot \mathbb{E}_{M \leftarrow E(n), M(T)}[r_S^E(T'|_M) - r_S^E(T|_M)|R] \\
 &\quad + \Pr_{M \leftarrow E(n)}[B] \cdot \mathbb{E}_{M \leftarrow E(n)}[r_S^E(T'|_M) - r_S^E(T|_M)|B] \\
 &\quad + \Pr_{M \leftarrow E(n), M(T)}[G] \cdot \mathbb{E}_{M \leftarrow E(n), M(T)}[r_S^E(T'|_M) - r_S^E(T|_M)|G]
 \end{aligned}$$

The first line restates Lemma 16. The second line simply observes that exactly one of the events  $R, B, G$  occur, and also uses linearity of expectation to take an expectation also over  $M(T)$ . The third line just observes that  $\mathbb{E}[Y \cdot I(X)] = \Pr[X] \cdot \mathbb{E}[Y|X]$  for any random variable  $Y$  and event  $X$ , breaks the sum into three parts (again using linearity of expectation), and observes that the event  $B$  is determined entirely by  $M$  and is independent of  $M(T)$ .

We now want to analyze the three terms separately. First, observe that by bullet three:

$$\mathbb{E}_{M \leftarrow E(n)}[r_S^E(T'|_M) - r_S^E(T|_M)|B] \leq c. \quad (1)$$

Next, observe that in either a good terminal event or a recursive event,  $(u, v) \notin M$ . Because  $T, T'$  are  $S$ -adjacent, this means that  $M(T), M(T')$  are identically distributed (and can therefore be coupled so that  $M(T) = M(T')$  with probability one), so the teams in  $T|_M$  and  $T'|_M$  are therefore the same. Finally, because  $T$  is an *independent* probabilistic tournament, and at least one team that participates in every match in  $M$  is eliminated,  $p_{ij}^{T|_M} = p_{ij}^T$  for all teams  $i, j$  which are present in  $T|_M$  (and also  $p_{ij}^{T'|_M} = p_{ij}^{T'}$ ). This means that  $T|_M \in \mathcal{S}_{\leq n-1}^\varepsilon$ , and also that  $T'|_M, T|_M$  are  $S$ -adjacent.

In a good terminal event, because  $\alpha_2^E(T|M) = 0$  by definition, we must therefore have:

$$\mathbb{E}_{M \leftarrow E(n), M(T)}[r_S^E(T'|M) - r_S^E(T|M)|G] \leq 0. \quad (2)$$

In a recursive event, we instead have:

$$\mathbb{E}_{M \leftarrow E(n), M(T)}[r_S^E(T'|M) - r_S^E(T|M)|R] \leq \alpha_2^E(\mathcal{S}_{\leq n-1}). \quad (3)$$

Finally, we may now use our inductive hypothesis and Equations (1), (2), (3) to conclude:

$$\begin{aligned} r_S^E(T') - r_S^E(T) &= \Pr_{M \leftarrow E(n), M(T)}[R] \cdot \mathbb{E}_{M \leftarrow E(n), M(T)}[r_S^E(T'|M) - r_S^E(T|M)|R] \\ &\quad + \Pr_{M \leftarrow E(n)}[B] \cdot \mathbb{E}_{M \leftarrow E(n)}[r_S^E(T'|M) - r_S^E(T|M)|B] \\ &\quad + \Pr_{M \leftarrow E(n), M(T)}[G] \cdot \mathbb{E}_{M \leftarrow E(n), M(T)}[r_S^E(T'|M) - r_S^E(T|M)|G] \\ &\leq \Pr_{M \leftarrow E(n), M(T)}[R] \cdot \frac{bc}{b+g} + \Pr_{M \leftarrow E(n)}[B] \cdot c + \Pr_{M \leftarrow E(n), M(T)}[G] \cdot 0 \\ &= \frac{bc}{b+g} + \Pr_{M \leftarrow E(n)}[B] \cdot \left(c - \frac{bc}{b+g}\right) + \Pr_{M \leftarrow E(n), M(T)}[G] \cdot \left(0 - \frac{bc}{b+g}\right) \\ &\leq (1 - b - g) \cdot \frac{bc}{b+g} + b \cdot c + g \cdot 0 = \frac{bc}{b+g} \end{aligned}$$

The first inequality just combines the work of Equations (1), (2), (3), and upper bounds the expected gains in all three cases using either the inductive hypothesis ( $R$ ), direct hypothesis ( $B$ ), or definition of good terminal event ( $G$ ). The final inequality observes that  $c \geq \frac{bc}{b+g} \geq 0$ , so the bound is maximized when  $B$  occurs as often as possible, while  $G$  occurs as little as possible (consistent with the hypotheses).

We have now shown that for any  $T \in \mathcal{S}^\varepsilon$ , and any  $T'$  which is  $S$ -adjacent, that  $r_S^E(T') - r_S^E(T) \leq \frac{bc}{b+g}$ . This establishes that  $\alpha_2^E(\mathcal{S}^\varepsilon) \leq \frac{bc}{b+g}$ . Proposition 9 extends this to  $\mathcal{T}^\varepsilon$ . ◀

Theorem 21 is our main framework for analysis. The remainder of this paper now computes the bounds required for the three bullets for the two tournament rules of interest as a function of  $\varepsilon$ , and substitutes to obtain tight bounds. Finally, it is worth briefly noting that the definition of  $\alpha_2^E(\mathcal{T}^\varepsilon)$  semantically assumes that teams must decide how to manipulate the outcome of their match *in advance*, before any matches are played. Still, any analysis that follows from Theorem 21 applies *even to manipulations which are decided upon as the match is played*. This is because the bound in bullet three of Theorem 21 must hold over *all*  $S$ -adjacent  $T'$ , not just one which was decided in advance. This claim is not central to our main results, and making it formal would be notationally cumbersome, so we provide only this brief discussion.

## 5 Warmup: Rederiving Bounds for Deterministic Tournaments

In this section, we rederive the main result of [14] (RSEB is 2-SNM-1/3), and analyze a novel tournament rule (RDM is 2-SNM-1/3) using a simple application of Theorem 21. This will require one further lemma about anonymous tournament rules to find additional base cases. The proof is in Appendix C.

► **Lemma 22.** *Let  $r$  be any anonymous tournament rule,  $S := \{u, v\}$ , and  $T, T'$  be independent probabilistic tournaments which are  $S$ -adjacent and satisfy  $p_{uw}^T = p_{vw}^T$  for all  $w \notin \{u, v\}$ . Then  $r_S(T) = r_S(T')$ .*



## 14:10 Approximately Strategyproof Tournament Rules in the Probabilistic Setting

We now prove that RDM is 2-SNM-1/3. Recall that  $\alpha_2(\mathcal{S}^{1/2}) = 1/3$ , so this is optimal.

► **Theorem 23.** *RDM is 2-SNM-1/3. Or in our language,  $\alpha_2^{RDM}(\mathcal{T}^{1/2}) = 1/3$ .*

**Proof.** Recall that we need to lower bound the probability of a good terminal event, upper bound the probability of a bad terminal event, and upper bound the gains from manipulation in case of a bad terminal event. For the deterministic case, the latter bound is particularly simple, and we will just observe that clearly  $\mathbb{E}_{M \leftarrow E(n)}[r_S^E(T'|_M) - r_S^E(T|_M)|B] \leq 1$ . This is simply because the maximum probability that any coalition can win in any tournament is 1, and the minimum is 0. So we have established that RDM satisfies  $c = 1$ .

To bound the probability of a bad terminal event, observe that a bad terminal event occurs only when  $u$  plays  $v$  in this round, which happens with probability exactly  $1/\binom{n}{2}$ . So RDM satisfies  $b = 1/\binom{n}{2}$ .

For the good terminal events, we claim that  $\Pr[G] \geq 2/\binom{n}{2}$ . First, observe that once we show this, we can plug into Theorem 21 and conclude  $\alpha_2^{RDM}(\mathcal{T}^{1/2}) \leq \frac{1 \cdot 1/\binom{n}{2}}{1/\binom{n}{2} + 2/\binom{n}{2}} = 1/3$ .

To see this bound, let  $\ell_u$  denote the number of teams which beat  $u$  but not  $v$ , and  $\ell_v$  denote the number of teams which beat  $v$  but not  $u$ . Without loss of generality let  $\ell_u \geq \ell_v$ .

If  $\ell_u + \ell_v = 0$ , then Lemma 22 already establishes that  $u$  and  $v$  can gain nothing by manipulating. If  $\ell_u + \ell_v \geq 2$ , then whenever  $u$  or  $v$  play a team which beat them, we have a good terminal event (because  $u$  or  $v$  is already eliminated before having ever played the  $(u, v)$  match, resulting in a base case). This happens with probability at least  $2/\binom{n}{2}$ , as desired.

If  $\ell_u + \ell_v = 1$ , then  $\ell_u = 1, \ell_v = 0$ . Let  $w$  be the unique team which beats  $u$  but not  $v$ . We claim that if  $w$  plays *either*  $u$  or  $v$  that we are in a good terminal event. Indeed, if  $w$  plays  $u$ , then  $u$  is eliminated having never played the  $(u, v)$  match. If instead  $w$  plays  $v$ , then  $w$  is eliminated, but now there are no remaining teams which beat  $u$  but not  $v$  (or vice versa) and Lemma 22 asserts that there are no further gains from manipulation. The probability that  $w$  plays  $u$  or  $v$  is  $2/\binom{n}{2}$ , as desired.

This handles all possible cases, and establishes that  $g \geq 2/\binom{n}{2}$  in all cases. Plugging into Theorem 21 as described above completes the proof. ◀

The analysis of RSEB is extremely similar to RDM, and requires only slightly more calculations to bound the probability of a good terminal event. This proof structure is fairly different than the original analysis in [14], and highlights the similarities to other recursive tournament rules. A complete proof of Theorem 24 appears in Appendix C.

► **Theorem 24** ([14]). *RSEB is 2-SNM-1/3. Or in our language,  $\alpha_2^{RSEB}(\mathcal{T}^{1/2}) = 1/3$ .*

We wrap up our warmup by highlighting key points of the analysis which will be relevant for our main results. First, observe that our analysis of both rules succeeded by simply upper-bounding  $c$  by 1. Improving this as a function of  $\varepsilon$  is the biggest technical difference between our warmup and main results. Second, observe that our analysis required Lemma 22 in case  $\ell_u < 2$ . In particular, we needed good terminal events *even when both  $u$  and  $v$  were not eliminated* (and this need continues in main results).

## 6 Optimal Bounds for Independent Probabilistic Tournaments

Before analyzing our two tournament rules, we extend the simple 3-team lower bound of [14] for  $\alpha_2(\mathcal{T}^{1/2})$  to  $\alpha_2(\mathcal{T}^\varepsilon)$ . The proof is in Appendix D.

► **Lemma 25.**  $\alpha_2(\mathcal{T}^\varepsilon) \geq \frac{1}{3}\varepsilon + \frac{2}{3}\varepsilon^2$ .



## 6.1 Gauntlets: Upper Bounding Gains from Bad Terminal Events

As previously noted, the main difference between our warmup and main results is bounding gain from bad terminal events. We provide a short, but key, structural insight about recursive tournament rules. Intuitively, a team  $u$  wins under rule  $r^E$  as long as they survive all elimination matches. Our key observation is that this defines a *gauntlet* of teams such that  $u$  wins if and only if they defeat every team in the gauntlet.

► **Definition 26 (Gauntlet).** *For deterministic tournament  $T$ , recursive tournament rule  $r$ , and team  $u$ , let  $T'$  be such that the outcome of the  $(v, w)$  match is the same for all  $v, w \neq u$ , but  $u$  is a Condorcet winner. The gauntlet for  $u$  in tournament  $T$  under recursive rule  $r$ ,  $G_u^r(T)$ , is the set of teams that  $u$  plays in elimination matches when  $r$  is executed on  $T'$ . If  $r$  is randomized, then  $G_u^r(T)$  is a random variable.*

*If  $T$  is a probabilistic tournament, we extend the notation  $G_u^r(T)$  to be the random variable which first samples  $T$ , then outputs  $G_u^r(T)$  (again over randomness in  $r$ ).*

For intuition, consider RSEB.  $u$  wins RSEB if and only if they win each of their  $\lceil \log_2(n) \rceil$  matches, so their gauntlet is a list of  $\lceil \log_2(n) \rceil$  teams, one per round. For RDM, however, the set of matches that  $u$  plays is itself a random variable (depending on how many times  $u$  is selected to play), but  $u$  still must win all these matches in order to win. For RKotH, the size of the gauntlet is a random variable as well. Importantly, however, observe that for all three rules (and any elimination rule), as soon as  $u$  loses a match, they are eliminated, so their gauntlet opponents can be set assuming that  $u$  won all previous matches.

Importantly, observe that  $u$ 's gauntlet does *not* depend on the outcome of any of its own matches (because  $T'$  immediately causes  $u$  to win all its matches anyway). This lets us make the following key observation, which requires  $E$  to be a matching elimination rule.

► **Lemma 27.** *Let  $r^E$  be an anonymous recursive tournament rule with a matching elimination rule,  $T$  be an independent probabilistic tournament on  $n$  teams,  $M \leftarrow E(n)$ , and  $u, v$  be two teams. Let also  $(u, v) \in M$  and  $w$  denote the winner of the  $(u, v)$  match. Then the random variable  $G_w^{r^E}(T|_M)$  is independent of  $w$ .*

**Proof.** Observe first that because  $M$  is a matching which contains  $(u, v)$ , that exactly one of  $\{u, v\}$  (namely,  $w$ ) is present in  $T|_M$ . Moreover, because  $T$  is independent probabilistic, the remaining teams in  $T|_M$  are independent of  $w$ . Because the definition of  $w$ 's gauntlet immediately considers a tournament  $T'$  which replaces the outcome of all matches involving  $w$  by having  $w$  be a Condorcet winner, the tournament  $T'$  is independent of  $w$  *except for whether  $w$  is labeled as “ $u$ ” or “ $v$ ”*. But because  $r^E(\cdot)$  is anonymous, its behavior on  $T'$  is independent of  $w$ 's label. This completes the proof. ◀

Note that Lemma 27 fails when  $M$  is not a matching. This is because: (a) perhaps both  $u$  and  $v$  are eliminated, and  $w$  is undefined, but also (b) even conditioned on  $w$  being defined, the set of teams in  $T|_M$  can depend on  $w$ . This is the main technical challenge in extending beyond matching elimination rules.<sup>5</sup>

<sup>5</sup> For example, in RKotH: conditioned on  $w := u$ , we know either that  $u$  was a Condorcet winner, or that  $v$  was selected to play everyone, so all teams which lose to  $v$  are eliminated. If instead  $w := v$ , we know either that  $v$  was a Condorcet winner, or that  $u$  was selected to play everyone. The teams which lose to  $u$  vs.  $v$  could be different, so the Lemma fails to hold.

► **Corollary 28.** Let  $r^E$  be an anonymous tournament rule with a matching elimination rule,  $T \in \mathcal{T}^\varepsilon$ ,  $S = \{u, v\}$  be any two teams, and  $T'$  be  $S$ -adjacent to  $T$ . Then:

$$\mathbb{E}_{M \leftarrow E(n)}[r_S^E(T'|M) - r_S^E(T|M)|B] \leq 2\varepsilon(\frac{1}{2} + \varepsilon).$$

**Proof.** Because we are in a bad terminal event, this means that  $(u, v) \in M$ . If we let  $w$  denote the winner of the  $(u, v)$  match in  $T$ , and  $w'$  denote the winner in  $T'$ , then we know that  $G_w^{r^E}(T|M)$  and  $G_{w'}^{r^E}(T'|M)$  are identically distributed by Lemma 27. This means that:

$$\begin{aligned} r_S^E(T|M) &= \mathbb{E}_{G_w^{r^E}(T|M)} \left[ \prod_{x \in G_w^{r^E}(T|M)} p_{wx}^T \right] \\ r_S^E(T'|M) &= \mathbb{E}_{G_{w'}^{r^E}(T'|M)} \left[ \prod_{x \in G_{w'}^{r^E}(T'|M)} p_{w'x}^{T'} \right] = \mathbb{E}_{G_w^{r^E}(T|M)} \left[ \prod_{x \in G_w^{r^E}(T|M)} p_{w'x}^T \right] \\ \Rightarrow r_S^E(T'|M) - r_S^E(T|M) &= \mathbb{E}_{G_w^{r^E}(T|M)} \left[ \prod_{x \in G_w^{r^E}(T|M)} p_{w'x}^T - \prod_{x \in G_w^{r^E}(T|M)} p_{wx}^T \right] \\ &\leq \mathbb{E}_{G_w^{r^E}(T|M)} \left[ \prod_{x \in G_w^{r^E}(T|M)} (\frac{1}{2} + \varepsilon) - \prod_{x \in G_w^{r^E}(T|M)} (\frac{1}{2} - \varepsilon) \right] \\ &= \mathbb{E}_{G_w^{r^E}(T|M)} \left[ (\frac{1}{2} + \varepsilon)^{|G_w^{r^E}(T|M)|} - (\frac{1}{2} - \varepsilon)^{|G_w^{r^E}(T|M)|} \right] \\ &\leq 2\varepsilon \end{aligned}$$

The first two lines follow by definition of the gauntlet, and Lemma 27. The third line is basic algebra. The fourth line follows as  $T \in \mathcal{T}^\varepsilon$ . The fifth line is again basic algebra, and the final line invokes Lemma 29, which is stated below (proof omitted from this version). The above calculations hold for any  $T|M, T'|M$ .

► **Lemma 29.** For all  $n \in \mathbb{N}_{\geq 0}$ , and  $\varepsilon \in [0, 1/2]$ :  $(\frac{1}{2} + \varepsilon)^n - (\frac{1}{2} - \varepsilon)^n \leq 2\varepsilon$ .

To see where the additional factor of  $(\frac{1}{2} + \varepsilon)$  comes from, recall that  $p_{uv}^T \in [1/2 - \varepsilon, 1/2 + \varepsilon]$ . Therefore,  $p_{uv}^{T'} - p_{uv}^T \leq \frac{1}{2} + \varepsilon$ . Consider now coupling the tournaments  $T$  and  $T'$  so the the result of every match except for the one between  $u$  and  $v$  is identical (this is possible because  $T$  and  $T'$  are independent probabilistic). Under this coupling,  $T$  and  $T'$  are identical with probability at least  $\frac{1}{2} - \varepsilon$  (and gains from manipulation are clearly only possible when  $T \neq T'$ , which occurs with probability at most  $\frac{1}{2} + \varepsilon$ , and independently of  $M$ ). Therefore,  $T \neq T'$  with probability at most  $(\frac{1}{2} + \varepsilon)$  and conditioned on this,  $r_S^E(T'|M) - r_S^E(T|M) \leq 2\varepsilon$ . ◀

## 6.2 RDM and RSEB

Corollary 28 is the main technical lemma to extend from  $\mathcal{T}^{1/2}$  to  $\mathcal{T}^\varepsilon$ . We begin with RDM.

► **Theorem 30.**  $\alpha_2^{RDM}(\mathcal{T}^\varepsilon) = \alpha_2(\mathcal{T}^\varepsilon) = \frac{1}{3}\varepsilon + \frac{2}{3}\varepsilon^2$ .

**Proof.** Let  $T \in \mathcal{S}^\varepsilon$ . Recall that we need to lower bound the probability of a good terminal event, upper bound the probability of a bad terminal event, and upper bound the gains from manipulation in case of a bad terminal event. We have already upper bounded the gains from a bad terminal event using Corollary 28 and can therefore take  $c = 2\varepsilon(\frac{1}{2} + \varepsilon)$ .

To bound the probability of a bad terminal event, observe that a bad terminal event occurs only when  $u$  plays  $v$  in this round, which happens with probability exactly  $1/\binom{n}{2}$ . So RDM satisfies  $b = 1/\binom{n}{2}$ .

For the good terminal events, we claim that  $\Pr[G] \geq 2/\binom{n}{2}$ . First, observe that once we show this, we can plug into Theorem 21 and conclude  $\alpha_2^{\text{RDM}}(\mathcal{T}^\varepsilon) \leq \frac{2\varepsilon(\frac{1}{2}+\varepsilon) \cdot 1/\binom{n}{2}}{1/\binom{n}{2} + 2/\binom{n}{2}} = \frac{1}{3}\varepsilon + \frac{2}{3}\varepsilon^2$ .

To see this bound, let  $\ell_u$  denote the number of teams which beat  $u$  with probability  $(1/2 + \varepsilon)$  but  $v$  with probability  $(1/2 - \varepsilon)$ , and  $\ell_v$  denote the number of teams which beat  $v$  with probability  $(1/2 + \varepsilon)$  but  $u$  with probability  $(1/2 - \varepsilon)$ . Without loss of generality let  $\ell_u \geq \ell_v$ .

If  $\ell_u = 0$ , then Lemma 22 already establishes that  $u$  and  $v$  can gain nothing by manipulating. If  $\ell_u + \ell_v \geq 2$ , then whenever  $u$  or  $v$  plays a team (not in  $\{u, v\}$ ) which beats them, we have a good terminal event (because  $u$  or  $v$  is already eliminated before having ever played the  $(u, v)$  match, so manipulating the match has no impact). Observe that any team which beats  $u$  (respectively,  $v$ ) with probability  $(1/2 + \varepsilon)$  beats  $v$  (respectively,  $u$ ) with probability at least  $(1/2 - \varepsilon)$ . Therefore, we have a good terminal event with probability at least  $2(1/2 + \varepsilon)/\binom{n}{2} + 2(1/2 - \varepsilon)/\binom{n}{2} = 2/\binom{n}{2}$ , as desired.<sup>6</sup>

If  $\ell_u = 1, \ell_v = 0$ , then let  $w$  be the unique team which beats  $u$  with probability  $(1/2 + \varepsilon)$  but  $v$  with probability  $(1/2 - \varepsilon)$ . We claim that if  $w$  plays *either*  $u$  or  $v$  that we have a good terminal event. Indeed, if  $w$  wins, then either  $u$  or  $v$  are eliminated having never played the  $(u, v)$  match. If instead  $w$  loses this match, then  $w$  is eliminated, but now  $p_{ux}^T = p_{vx}^T$  for all remaining teams  $x$ , and Lemma 22 asserts that there are no further gains from manipulation. The probability that  $w$  plays  $u$  or  $v$  is  $2/\binom{n}{2}$ , as desired.

This handles all possible cases, and establishes that  $g \geq 2/\binom{n}{2}$  in all cases. Plugging into Theorem 21 as described above completes the proof.  $\blacktriangleleft$

The analysis for RSEB is again similar to RDM, but some calculations are more involved.

► **Theorem 31.**  $\alpha_2^{\text{RSEB}}(\mathcal{T}^\varepsilon) = \alpha_2(\mathcal{T}^\varepsilon) = \frac{1}{3}\varepsilon + \frac{2}{3}\varepsilon^2$ .

**Proof.** Let  $T \in \mathcal{S}^\varepsilon$ . We have already upper bounded the gains from a bad terminal event using Corollary 28 and can therefore take  $c = 2\varepsilon(\frac{1}{2} + \varepsilon)$ .

To bound the probability of a bad terminal event, observe that a bad terminal event occurs only when  $u$  plays  $v$  this round, which happens with probability exactly  $1/(n' - 1)$ .<sup>7</sup> So RSEB satisfies  $b = 1/(n' - 1)$ .

For the good terminal events, we claim that  $\Pr[G] \geq 2/(n' - 1)$ . First, observe that once we show this, we can plug into Theorem 21 and conclude  $\alpha_2^{\text{RSEB}}(\mathcal{T}^\varepsilon) \leq \frac{2\varepsilon(\frac{1}{2}+\varepsilon) \cdot 1/(n'-1)}{1/(n'-1) + 2/(n'-1)} = \frac{1}{3}\varepsilon + \frac{2}{3}\varepsilon^2$ .

To see this bound (which requires more calculations than previous proofs), let  $\ell_u$  denote the number of teams which beat  $u$  with probability  $(1/2 + \varepsilon)$  but  $v$  with probability  $(1/2 - \varepsilon)$ , and  $\ell_v$  denote the number of teams which beat  $v$  with probability  $(1/2 + \varepsilon)$  but  $u$  with probability  $(1/2 - \varepsilon)$ . Without loss of generality let  $\ell_u \geq \ell_v$ .

**Case One:**  $\ell_u + \ell_v \leq 1$ . If  $\ell_u = 0$ , then Lemma 22 already establishes that  $u$  and  $v$  can gain nothing by manipulating. If  $\ell_u = 1, \ell_v = 0$ , then let  $w$  be the unique team which beats  $u$  with probability  $(1/2 + \varepsilon)$  but  $v$  with probability  $(1/2 - \varepsilon)$ . We claim that if  $w$

<sup>6</sup> To be extra clear, the two matches  $(w, u)$ ,  $(w, v)$  together contribute probability  $1/\binom{n}{2}$  to the probability of a good terminal event, as long as  $w$  beats either  $u$  or  $v$  with probability  $(1/2 + \varepsilon)$ . Because there are two such teams, we get this twice.

<sup>7</sup> Recall in RSEB that  $n'$  denotes the total number of teams plus dummy teams, and is  $2^{\lceil \log_2(n) \rceil}$

plays *either*  $u$  or  $v$  that we are in a good terminal event. Indeed, if  $w$  wins this match, then either  $u$  or  $v$  are eliminated having never played the  $(u, v)$  match. If instead  $w$  loses this match, then  $w$  is eliminated, but now  $p_{ux}^T = p_{vx}^T$  for all remaining teams  $x$ , and Lemma 22 asserts that there are no further gains from manipulation. The probability that  $w$  plays  $u$  or  $v$  is  $2/(n' - 1)$ , as desired.

**Case Two:**  $\ell_u + \ell_v = 2$ . Next, consider the case where  $\ell_u + \ell_v = 2$ , and call the two relevant teams  $x, w$ . Observe first that if  $u$  plays  $x$  and  $v$  plays  $w$ , or if  $u$  plays  $w$  and  $v$  plays  $x$ , then we are surely in a good terminal event. This is because either (a)  $u$  or  $v$  is eliminated without having played the  $(u, v)$  match, or (b) *both*  $x$  and  $w$  are eliminated (allowing us to invoke Lemma 22). This occurs with probability  $\frac{2}{(n'-1)(n'-3)}$ . There are also the cases where exactly one of  $\{w, x\}$  plays a team in  $\{u, v\}$ . For any given pair  $(a, b)$ , with  $a \in \{w, x\}$  and  $b \in \{u, v\}$ , this case occurs with probability  $\frac{(n'-4)}{(n'-1)(n'-3)}$ .<sup>8</sup> Two of these cases contribute at least a  $(1/2 + \varepsilon)$  probability of eliminating the team in  $\{u, v\}$ , and the other two contribute at least a  $(1/2 - \varepsilon)$  probability. So in total, all four cases contribute at least  $\frac{2(n'-4)}{(n'-1)(n'-3)}$ , and together we get that a good terminal event occurs with probability at least:

$$\frac{2}{(n'-1)(n'-3)} + \frac{2(n'-4)}{(n'-1)(n'-3)} = \frac{2(n'-3)}{(n'-1)(n'-3)} = \frac{2}{n'-1}.$$

**Case Three:**  $\ell_u + \ell_v \geq 3$ . Next, consider the case where  $\ell_u + \ell_v \geq 3$  (observe that this implies  $n \geq 5$ ). We will show that either  $u$  or  $v$  are eliminated with probability at least  $2/(n' - 1)$ . Indeed, let  $L_u$  denote the set of teams which beat  $u$  with probability  $(1/2 + \varepsilon)$  (but not  $v$ ), and  $L_v$  denote the set of teams which beat  $v$  with probability  $(1/2 + \varepsilon)$  (but not  $u$ ). Then consider the case where  $u$  plays a team in  $L_u$ , or  $v$  plays a team in  $L_v$ . Conditioned on this, *both*  $u$  and  $v$  survive with probability at most  $(1/2 + \varepsilon)(1/2 - \varepsilon) \leq 1/4$ , so one of  $\{u, v\}$  is eliminated with probability at least  $3/4$ . So one sufficient condition would be to establish that  $u$  plays a team in  $L_u$  or  $v$  plays a team in  $L_v$  with probability at least  $\frac{8/3}{n'-1}$  (because conditioned on this, one of  $\{u, v\}$  is eliminated with probability  $3/4$ , for a total probability of at least  $\frac{2}{n'-1}$  that one of  $\{u, v\}$  is eliminated). In the subcase that  $\ell_u \geq 3$ , then the probability is in fact at least  $3/(n' - 1)$ , as desired. Clearly, this probability is monotone in  $\ell_u, \ell_v$ , so this leaves the only remaining subcase as  $\ell_u = 2, \ell_v = 1$ .

**Subcase Three-A:**  $\ell_u = 2, \ell_v = 1, n \geq 9$ . For the subcase of  $\ell_u = 2, \ell_v = 1$ , the probability that  $u$  plays a team in  $L_u$  or  $v$  plays a team in  $L_v$  is  $\frac{2}{n'-1} + \frac{(n'-5)}{(n'-1)(n'-3)}$ .<sup>9</sup> As  $n' \geq 16$  (because  $n \geq 9$  and  $n'$  is a power of 2), we have that  $(n' - 5)/(n' - 3) \geq 11/13$ , meaning that  $\frac{2}{n'-1} + \frac{(n'-5)}{(n'-1)(n'-3)} \geq \frac{37/13}{n'-1} > \frac{8/3}{n'-1}$ , which resolves this case by the work above.

**Subcase Three-B:**  $\ell_u = 2, \ell_v = 1, \varepsilon \geq \frac{1}{2\sqrt{13}}, n \in [5, 8]$ . When  $n \in [5, 8]$ , we have  $n' = 8$ .

This means that  $(n' - 5)/(n' - 3) = 3/5$ , and therefore  $\frac{2}{n'-1} + \frac{(n'-5)}{(n'-1)(n'-3)} = \frac{13/5}{n'-1}$ .

Unfortunately, this is  $< \frac{8/3}{n'-1}$ , meaning that this case doesn't immediately resolve by the method above. In this range, we do an explicit case analysis. First, observe that  $u$  and  $v$  are *least* likely to be eliminated when there are more dummy teams (because  $u$  and  $v$

<sup>8</sup> Because the probability that  $a$  plays  $b$  is  $1/(n' - 1)$ , and the probability that the other two teams do not play, conditioned on this, is  $(n' - 4)/(n' - 3)$ .

<sup>9</sup> To see this, observe that the probability that  $u$  plays a team in  $L_u$  is  $2/(n' - 1)$ . The probability that  $u$  plays a team not in  $L_u \cup L_v \cup \{v\}$  is  $(n' - 5)/(n' - 1)$ . Conditioned on this, the probability that  $v$  plays the team in  $L_v$  is  $1/(n' - 3)$  (and this is the only way that  $v$  can possibly play the team in  $L_v$  without  $u$  playing  $L_u$ ).

beat dummy teams with probability one, but real teams with probability at most  $1/2 + \varepsilon$ ). So the worst case to consider is when  $n = 5$ : two real teams beat only  $u$ , one beats only  $v$ , and there are three dummy teams. We just have to compute several cases:

- Perhaps  $u$  plays one of the  $\ell_u$  teams, and  $v$  plays one of the  $\ell_v$  teams. This occurs with probability  $\frac{2}{7} \cdot \frac{1}{5}$ , and eliminates  $u$  or  $v$  with probability  $1 - (1/2 - \varepsilon)^2 = 3/4 + \varepsilon - \varepsilon^2$ .
- Perhaps  $u$  plays one of the  $\ell_u$  teams, and  $v$  plays the other. This occurs with probability  $\frac{2}{7} \cdot \frac{1}{5}$ , and eliminates  $u$  or  $v$  with probability  $1 - (1/2 + \varepsilon)(1/2 - \varepsilon) = 3/4 + \varepsilon^2$ .
- Perhaps  $u$  plays one of the  $\ell_u$  teams, and  $v$  plays a dummy team. This occurs with probability  $\frac{2}{7} \cdot \frac{3}{5}$ , and eliminates  $u$  with probability  $1/2 + \varepsilon$ .
- Perhaps  $u$  plays one of the  $\ell_v$  teams, and  $v$  plays one of the  $\ell_u$  teams. This occurs with probability  $\frac{1}{7} \cdot \frac{2}{5}$ , and eliminates  $u$  or  $v$  with probability  $1 - (1/2 + \varepsilon)^2 = 3/4 - \varepsilon - \varepsilon^2$ .
- Perhaps  $u$  plays one of the  $\ell_v$  teams, and  $v$  plays a dummy team. This occurs with probability  $\frac{1}{7} \cdot \frac{3}{5}$ , and eliminates  $u$  with probability  $1/2 - \varepsilon$ .
- Perhaps  $u$  plays a dummy team, and  $v$  plays one of the  $\ell_v$  teams. This occurs with probability  $\frac{3}{7} \cdot \frac{1}{5}$ , and eliminates  $v$  with probability  $1/2 + \varepsilon$ .
- Perhaps  $u$  plays a dummy team, and  $v$  plays one of the  $\ell_u$  teams. This occurs with probability  $\frac{3}{7} \cdot \frac{2}{5}$ , and eliminates  $v$  with probability  $1/2 - \varepsilon$ .
- Perhaps both  $u$  and  $v$  play dummy teams. This happens with probability  $\frac{3}{7} \cdot \frac{2}{5}$ , but eliminates neither team.

So either  $u$  or  $v$  is eliminated (without playing each other) with probability:

$$\begin{aligned} & \frac{2}{35} \cdot (3/4 + \varepsilon - \varepsilon^2) + \frac{2}{35} \cdot (3/4 + \varepsilon^2) + \frac{6}{35} \cdot (1/2 + \varepsilon) + \frac{2}{35} \cdot (3/4 - \varepsilon - \varepsilon^2) \\ & + \frac{3}{35} \cdot (1/2 - \varepsilon) + \frac{3}{35} \cdot (1/2 + \varepsilon) + \frac{6}{35} \cdot (1/2 - \varepsilon) \\ & = \frac{27}{70} - \frac{2}{35}\varepsilon^2 \geq \frac{26}{70} > 2/7. \end{aligned}$$

Note that when  $n \in [5, 8]$ , the probability of a bad event is exactly  $1/7$ , and the above work establishes that in case Three-B, the probability of a good terminal event is at least twice that of a bad terminal event.

The arguments above handle all possible cases, and establishes that  $g \geq 2/(n' - 1)$  in all cases. Plugging into Theorem 21 as described above completes the proof. ◀

## 7 Conclusion and Open Problems

We take a beyond worst-case view on manipulability of tournament rules, and nail down optimal guarantees as a function of the uncertainty of match outcomes. Specifically, our main result shows that  $\alpha_2(\mathcal{T}^\varepsilon) = \varepsilon/3 + 2\varepsilon^2/3$ , and this is achieved *for all*  $\varepsilon$  by Randomized Death Match and Randomized Single Elimination Bracket. Our main technical contribution is a framework to analyze recursive tournament rules.

There are two natural directions for future work. The first concerns further work in the probabilistic setting: does  $\alpha_2^{\text{RKotH}}(\mathcal{T}^\varepsilon) = \varepsilon/3 + 2\varepsilon^2/3$ ? The main technical barrier is replacing Lemma 27, which only holds for matching elimination rules.<sup>10</sup> In addition, it is interesting to analyze probabilistic tournaments which are not independent. Here there are technical barriers to overcome (many of our steps do require independence), but also conceptual

<sup>10</sup>There are other barriers to using our precise definitions, but these barriers seem semantic rather than substantial.

ones: if tournament match outcomes are correlated, should we consider manipulations which are correlated with external outcomes as well? If so, is there a natural way to consider manipulations which are “not more correlated than the original tournament itself”?

A second direction concerns applications of our recursive framework towards other problems in approximately strategy-proof tournament design. For example, it is still an open question following [15] what is  $\alpha_k(\mathcal{T}^{1/2})$  for *any*  $k > 2$ . Our recursive framework may prove useful for analyzing this, or at least determining achievable guarantees for recursive rules.

---

## References

---

- 1 Alon Altman and Robert Kleinberg. Nonmanipulable randomized tournament selections. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11-15, 2010*. AAAI Press, 2010. URL: <http://www.aaai.org/ocs/index.php/AAAI/AAAI10/paper/view/1703>.
- 2 Alon Altman, Ariel D. Procaccia, and Moshe Tennenholtz. Nonmanipulable selections from a tournament. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence, IJCAI’09*, pages 27–32, San Francisco, CA, USA, 2009. Morgan Kaufmann Publishers Inc. URL: <http://dl.acm.org/citation.cfm?id=1661445.1661451>.
- 3 J. S. Banks. Sophisticated voting outcomes and agenda control. *Social Choice and Welfare*, 1(4):295–306, December 1985. doi:10.1007/BF00649265.
- 4 Felix Brandt, Vincent Conitzer, Ulle Endriss, Jérôme Lang, and Ariel D Procaccia. *Handbook of Computational Social Choice*. Cambridge University Press, 2016.
- 5 Laszlo Csato. European qualifiers 2018 FIFA world cup qualification can be manipulated, September 2017. URL: <https://mpra.ub.uni-muenchen.de/id/eprint/83437>.
- 6 Peter C. Fishburn. Condorcet social choice functions. *SIAM Journal on Applied Mathematics*, 33(3):469–489, 1977. doi:10.1137/0133030.
- 7 David C. Fisher and Jennifer Ryan. Optimal strategies for a generalized “scissors, paper, and stone” game. *The American Mathematical Monthly*, 99(10):935–942, 1992. doi:10.1080/00029890.1992.11995957.
- 8 Michael P. Kim, Warut Suksompong, and Virginia Vassilevska Williams. Who can win a single-elimination tournament? In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA.*, pages 516–522, 2016. URL: <http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/12194>.
- 9 Michael P. Kim and Virginia Vassilevska Williams. Fixing tournaments for kings, chokers, and more. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pages 561–567, 2015. URL: <http://ijcai.org/Abstract/15/085>.
- 10 G. Laffond, J.F. Laslier, and M. Le Breton. The bipartisan set of a tournament game. *Games and Economic Behavior*, 5(1):182–201, 1993. doi:10.1006/game.1993.1010.
- 11 J.F. Laslier. *Tournament Solutions and Majority Voting*. Studies in Economic Theory (Berlin, Germany), 7. Springer, 1997. URL: <https://books.google.com/books?id=vYbGAAAAIAAJ>.
- 12 Nicholas R. Miller. A new solution set for tournaments and majority voting: Further graph-theoretical approaches to the theory of voting. *American Journal of Political Science*, 24(1):68–96, 1980. URL: <http://www.jstor.org/stable/2110925>.
- 13 Marc Pauly. Can strategizing in round-robin subtournaments be avoided? *Social Choice and Welfare*, 43(1):29–46, 2014. URL: <http://EconPapers.repec.org/RePEc:spr:sochwe:v:43:y:2014:i:1:p:29-46>.
- 14 Jon Schneider, Ariel Schwartzman, and S. Matthew Weinberg. Condorcet-consistent and approximately strategyproof tournament rules. In *8th Innovations in Theoretical Computer Science Conference, ITCS 2017, January 9-11, 2017, Berkeley, CA, USA*, pages 35:1–35:20, 2017. doi:10.4230/LIPIcs.ITCS.2017.35.



- 15 Ariel Schwartzman, S. Matthew Weinberg, Eitan Zlatin, and Albert Zuo. Approximately strategyproof tournament rules: On large manipulating sets and cover-consistence. In Thomas Vidick, editor, *11th Innovations in Theoretical Computer Science Conference, ITCS 2020, January 12-14, 2020, Seattle, Washington, USA*, volume 151 of *LIPIcs*, pages 3:1–3:25. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.ITCS.2020.3.
- 16 Kenneth A. Shepsle and Barry R. Weingast. Uncovered sets and sophisticated voting outcomes with implications for agenda institutions. *American Journal of Political Science*, 28(1):49–74, 1984. URL: <http://www.jstor.org/stable/2110787>.
- 17 Isabelle Stanton and Virginia Vassilevska Williams. Rigging tournament brackets for weaker players. In *IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16-22, 2011*, pages 357–364, 2011. doi:10.5591/978-1-57735-516-8/IJCAI11-069.
- 18 Virginia Vassilevska Williams. Fixing a tournament. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11-15, 2010*, 2010. URL: <http://www.aaai.org/ocs/index.php/AAAI/AAAI10/paper/view/1726>.

## A Omitted Proofs from Section 2

**Proof of Proposition 9.** The second statement of the lemma clearly follows from the first. It is trivial to see that  $\alpha_k^r(\mathcal{T}^\varepsilon) \geq \alpha_k^r(\mathcal{S}^\varepsilon)$ , as  $\mathcal{S}^\varepsilon \subseteq \mathcal{T}^\varepsilon$ . To get intuition for why  $\alpha_k^r(\mathcal{T}) \leq \alpha_k^r(\mathcal{S}^\varepsilon)$ , observe that every probabilistic tournament in  $\mathcal{T}^\varepsilon$  can be written as a convex combination of probabilistic tournaments in  $\mathcal{S}^\varepsilon$ . Indeed, this intuition drives the proof.

To see this formally, let  $T$  be an arbitrary (independent probabilistic) tournament in  $\mathcal{T}^\varepsilon$ , and let  $T'$  be any  $S$ -adjacent (independent probabilistic) tournament for some  $|S| \leq k$ . Then  $p_{ij}^T \in [1/2 - \varepsilon, 1/2 + \varepsilon]$  for all  $i, j$ , by definition. Consider the following procedure to jointly sample tournaments from  $T, T'$ . The procedure first constructs another pair of probabilistic tournaments  $U, U'$ , and then samples from these.

1. For all  $(i, j)$ , set  $q_{ij} := \frac{p_{ij}^T - (1/2 - \varepsilon)}{2\varepsilon}$ . Observe that  $q_{ij} \cdot (1/2 + \varepsilon) + (1 - q_{ij}) \cdot (1/2 - \varepsilon) = p_{ij}^T$ , and that  $q_{ij} \in [0, 1]$  as  $p_{ij}^T \in [1/2 - \varepsilon, 1/2 + \varepsilon]$ .
2. For each  $(i, j)$ , independently, set  $p_{ij}$  equal to  $(1/2 + \varepsilon)$  with probability  $q_{ij}$  and equal to  $(1/2 - \varepsilon)$  with probability  $(1 - q_{ij})$ .
3. For all  $(i, j)$ , set  $p_{ij}^U := p_{ij}$ .
4. For all  $(i, j)$  such that  $\{i, j\} \not\subseteq S$ , set  $p^{U'} := p^U$ . For other  $(i, j)$ , set  $p_{ij}^{U'} := p_{ij}^{T'}$ .
5. Draw a pair of tournaments according to  $U, U'$  (independently, say).

Observe that this procedure correctly draws two tournament according to  $T, T'$ . Indeed, it is easy to see for each output tournament that the outcome of each match is independent, simply because they are independent in  $U$  (resp.  $U'$ ), and because the random variables  $p_{ij}^U$  (resp.  $p_{ij}^{U'}$ ) are independent. Moreover, we claim that the probability that  $i$  beats  $j$  is exactly  $p_{ij}^T$ . Indeed, the probability that  $i$  beats  $j$  in  $U$  is:  $\mathbb{E}[p_{ij}^U] = q_{ij} \cdot (1/2 + \varepsilon) + (1 - q_{ij}) \cdot (1/2 - \varepsilon) = p_{ij}^T$ . In  $U'$ , when  $\{i, j\} \subseteq S$ , the probability that  $i$  beats  $j$  in  $U'$  is clearly  $p_{ij}^{T'}$  by definition. When  $\{i, j\} \not\subseteq S$ , the probability that  $i$  beats  $j$  is also  $p_{ij}^T$ , which is equal to  $p_{ij}^{T'}$  as  $T, T'$  are  $S$ -adjacent. Additionally, observe that: (a)  $U \in \mathcal{S}^\varepsilon$  and (b)  $U$  and  $U'$  are  $S$ -adjacent. Now, consider any tournament rule  $r(\cdot)$ :

$$\begin{aligned} \text{For all } i: r_i(T) &= \mathbb{E}[r_i(U)], \text{ and } r_i(T') = \mathbb{E}[r_i(U')] \\ \Rightarrow r_S(T') - r_S(T) &= \mathbb{E}[r_S(U') - r_S(U)] \leq \alpha_k^r(\mathcal{S}^\varepsilon) \end{aligned}$$

Indeed, the first line is simply linearity of expectation, once the previous work confirms that going through  $U, U'$  is a valid way to draw tournaments from  $T, T'$ . The third line then also follows by linearity of expectation. The final line follows as  $U \in \mathcal{S}^\varepsilon$ , and  $U, U'$  are  $S$ -adjacent. This completes the proof, as we have now shown that  $\alpha_k^r(\mathcal{T}^\varepsilon) \leq \alpha_k^r(\mathcal{S}^\varepsilon)$ .  $\blacktriangleleft$



## B

 Omitted Proofs from Section 3

**Proof of Lemma 16.** This follows immediately from linearity of expectation. For all  $i$ , the probability that  $i$  wins in tournament  $T$  under rule  $r^E(\cdot)$  is the expected probability that  $i$  wins in  $T|_M$ . Summing over  $i \in S$ , and repeating this for  $T'$  yields the lemma. ◀

## C

 Omitted Proofs from Section 5

**Proof of Lemma 22.** Let  $\sigma(\cdot)$  denote the permutation which swaps  $u$  and  $v$ . Consider any two deterministic tournaments  $U, \sigma(U)$ . Then because  $r(\cdot)$  is anonymous, we have:

$$\begin{aligned} r_u(U) + r_v(U) &= r_{\sigma(u)}(\sigma(U)) + r_{\sigma(v)}(\sigma(U)) = r_v(\sigma(U)) + r_u(\sigma(U)) \\ &\Rightarrow r_S(U) = r_S(\sigma(U)) \end{aligned}$$

Indeed, the first line simply applies anonymity, and the second line simply applies  $\sigma$ . Now let's return to  $T, T'$  (which are independent probabilistic tournaments, rather than deterministic). Consider the following process to draw  $T$  and  $T'$  jointly:

1. To emphasize that  $p_{uw}^T = p_{vw}^T$ , for all  $w \notin \{u, v\}$ , denote by  $p_w^T := p_{uw}^T$ .
2. Without loss of generality, let  $p_{uv}^T \leq p_{uv}^{T'}$ .
3. Draw the outcome of all matches involving two teams both  $\notin \{u, v\}$ . Set the outcome of these matches the same for  $T$  and  $T'$ .
4. For all  $w \notin \{u, v\}$ , draw  $q_{w1}$  and  $q_{w2}$  iid and uniformly from  $[0, 1]$ . Draw  $q_{uv}$  independently and uniformly from  $[0, 1]$ .
5. For all  $w \notin \{u, v\}$ , have  $u$  beat  $w$  if and only if  $q_{w1} < p_w^T$ . Have  $v$  beat  $w$  if and only if  $q_{w2} < p_w^T$ . Have  $u$  beat  $v$  if and only if  $q_{uv} < p_{uv}^T$ .
6. If  $q_{uv} \notin [p_{uv}^T, p_{uv}^{T'}]$ , set  $T' := T$ .
7. If  $q_{uv} \in [p_{uv}^T, p_{uv}^{T'}]$ , then set  $T' := \sigma(T)$ .

This process clearly satisfies that with probability one, either  $T' = T$  or  $T' = \sigma(T)$ . By the work above, this means that  $r_S(T) = r_S(T')$  as desired, as long as we confirm that this process validly samples both  $T$  and  $T'$ . It is easy to see that the process is valid for  $T$ : the match outcomes are clearly independent, and any team  $w$  beats  $x$  if and only if a uniformly random draw from  $[0, 1]$  is  $< p_{wx}^T$  (which happens with probability  $p_{wx}^T$ , as desired).

To see that the process is valid for  $T'$ , observe first that  $u$  beats  $v$  with probability exactly  $p_{uv}^{T'}$ , because  $u$  beats  $v$  whenever  $q_{uv} < p_{uv}^{T'}$ . Moreover, after conditioning on  $q_{uv}$ , the outcome of each  $(u, w)$  match is either set according to  $q_{w1}$  (an independent, uniform draw from  $[0, 1]$ ), or  $q_{w2}$  (also an independent, uniform draw from  $[0, 1]$ ). In either case, these match results are all set independently and with the correct probability (because  $T, T'$  are  $S$ -adjacent, and because  $q_{w1}, q_{w2}$  are iid because  $p_{uw}^T = p_{vw}^T$  for all  $w$ ). This completes the proof. ◀

**Proof of Theorem 24.** We again simply let  $c = 1$  and take the trivial bound on the gain in bad terminal events.

To bound the probability of a bad terminal event, observe that a bad terminal event occurs only when  $u$  plays  $v$  this round, which happens with probability exactly  $1/(n' - 1)$ .<sup>11</sup> So RSEB satisfies  $b = 1/(n' - 1)$ .

For the good terminal events, we claim that  $\Pr[G] \geq 2/(n' - 1)$ . First, observe that once we show this, we can plug into Theorem 21 and conclude  $\alpha_2^{\text{RSEB}}(\mathcal{T}^{1/2}) \leq \frac{1 \cdot 1/(n' - 1)}{1/(n' - 1) + 2/(n' - 1)} = 1/3$ .

<sup>11</sup> Recall that in RSEB,  $n' := 2^{\lceil \log_2(n) \rceil}$ .

To establish this bound, let  $\ell_u$  denote the number of teams which beat  $u$  but not  $v$ ,  $\ell_v$  denote the number of teams which beat  $v$  but not  $u$ . Without loss of generality let  $\ell_u \geq \ell_v$ .

If  $\ell_u + \ell_v = 0$ , then Lemma 22 already establishes that  $u$  and  $v$  gain nothing by manipulating. If  $\ell_u \geq 2$ , then whenever  $u$  plays a team which beats them, we have a good terminal event (because  $u$  is already eliminated before having ever played the  $(u, v)$  match, so manipulating the match has no impact). This happens with probability at least  $2/(n' - 1)$ .

If  $\ell_u = 1, \ell_v = 0$ , then let  $w$  denote the unique team which beats  $u$  but not  $v$ . We now claim that if  $w$  plays *either*  $u$  or  $v$  that we are in a good terminal event. Indeed, if  $w$  plays  $u$ , then  $u$  is eliminated having never played the  $(u, v)$  match. If instead  $w$  plays  $v$ , then  $w$  is eliminated, but now there are no remaining teams which beat  $u$  but not  $v$  (or vice versa) and Lemma 22 asserts that there are no further gains from manipulation. The probability that  $w$  plays  $u$  or  $v$  is  $2/(n' - 1)$ , as desired.

Finally, if  $\ell_u = \ell_v = 1$ , let  $w$  denote the unique team which beats  $u$  but not  $v$ , and  $x$  denote the unique team which beats  $v$  but not  $u$  (this case requires slightly more calculations than RDM). If  $u$  plays  $w$ , or  $v$  plays  $x$ , then at least one of  $u, v$  is eliminated before the  $(u, v)$  match is played, and therefore no gains are possible. This happens with probability  $2/(n' - 1) - \frac{1}{(n' - 1)(n' - 3)}$ . In addition, if  $u$  plays  $x$  and  $v$  plays  $w$ , then both  $x$  and  $w$  will be eliminated, and Lemma 22 asserts that there are no further gains from manipulation. This occurs with probability  $\frac{1}{(n' - 1)(n' - 3)}$ . Therefore, a good terminal event happens with probability at least  $2/(n' - 1)$ , as desired. This handles all possible cases, and establishes that  $g \geq 2/(n' - 1)$  in all cases. Plugging into Theorem 21 as described completes the proof. ◀

## D Omitted Proofs from Section 6

**Proof of Lemma 25.** Consider a 3-player tournament  $T$  with teams  $u, v$ , and  $w$ ,  $p_{uv}^T = p_{vw}^T = p_{wu}^T = \frac{1}{2} + \varepsilon$ .<sup>12</sup> Noting that  $T$  is randomized, there are two possible types of deterministic outcomes: outcomes where there is a Condorcet winner, and outcomes which form a cycle (either  $v$  beats  $u$ ,  $u$  beats  $w$ , and  $w$  beats  $v$ , or vice versa).

In  $T$ , each of the three players has probability  $(\frac{1}{2} + \varepsilon)(\frac{1}{2} - \varepsilon)$  of being a Condorcet winner.

Call the cycle where  $v$  beats  $u$  “cycle 1”; this occurs with probability  $(\frac{1}{2} + \varepsilon)^3$ . Call the opposing cycle (where  $u$  beats  $v$ ) “cycle 2”; this occurs with probability  $(\frac{1}{2} - \varepsilon)^3$ .

Let  $r$  be any Condorcet-consistent tournament rule. Denote by  $\gamma_x$  the probability that the rule selects  $x$  as the winner when cycle 1 occurs (for any  $x \in \{u, v, w\}$ ). Denote by  $\beta_x$  the probability that the rule selects  $x$  as the winner when cycle 2 occurs. Recall that  $r$  must select  $x$  as the winner with probability 1 when  $x$  is the Condorcet winner.

Suppose that  $u$  and  $v$  collude so that  $u$  throws their match to  $v$ . Specifically, let  $T^{uv}$  denote the  $\{u, v\}$ -adjacent tournament to  $T$  where  $p_{uv}^{T^{uv}} = 0$  (instead of  $1/2 + \varepsilon$ ). In  $T^{uv}$ ,  $v$  is a Condorcet winner with probability  $\frac{1}{2} + \varepsilon$  (they need only beat  $w$ ), cycle 2 occurs with probability  $(\frac{1}{2} - \varepsilon)^2$  ( $v$  must lose to  $w$ , who must lose to  $u$ ), and  $w$  is a Condorcet winner with probability  $(\frac{1}{2} + \varepsilon)(\frac{1}{2} - \varepsilon)$ . No other outcomes are possible. We then have:

$$\begin{aligned} r_u(T^{uv}) + r_v(T^{uv}) - r_u(T) - r_v(T) &= (\tfrac{1}{2} + \varepsilon) + (\beta_u + \beta_v) \cdot (\tfrac{1}{2} - \varepsilon)^2 \\ &\quad - 2(\tfrac{1}{2} + \varepsilon)(\tfrac{1}{2} - \varepsilon) - (\beta_u + \beta_v) \cdot (\tfrac{1}{2} - \varepsilon)^3 - (\gamma_u + \gamma_v)(\tfrac{1}{2} + \varepsilon)^3 \\ &= 2\varepsilon \cdot (\tfrac{1}{2} + \varepsilon) + (\beta_u + \beta_v) \cdot ((\tfrac{1}{2} - \varepsilon)^2 - (\tfrac{1}{2} - \varepsilon)^3) - (\gamma_u + \gamma_v)(\tfrac{1}{2} + \varepsilon)^3 \end{aligned}$$

<sup>12</sup>That is, if  $\varepsilon = 1/2$ , this is the same 3-cycle example from [14].

**14:20    Approximately Strategyproof Tournament Rules in the Probabilistic Setting**

Identical calculations hold for  $T^{vw}, T^{wu}$ . We can sum these three together to achieve:

$$\begin{aligned} & (r_u(T^{uv}) + r_v(T^{uv}) - r_u(T) - r_v(T)) + (r_v(T^{vw}) + r_w(T^{vw}) - r_v(T) - r_w(T)) \\ & \quad + (r_w(T^{wu}) + r_u(T^{wu}) - r_w(T) - r_u(T)) \\ &= 6\varepsilon \cdot (\tfrac{1}{2} + \varepsilon) + 2(\beta_u + \beta_v + \beta_w) \cdot ((\tfrac{1}{2} - \varepsilon)^2 - (\tfrac{1}{2} - \varepsilon)^3) - 2(\gamma_u + \gamma_v + \gamma_w)(\tfrac{1}{2} + \varepsilon)^3 \\ &= 6\varepsilon \cdot (\tfrac{1}{2} + \varepsilon) + 2 \cdot (\tfrac{1}{2} - \varepsilon)^2 \cdot (\tfrac{1}{2} + \varepsilon) - 2(\tfrac{1}{2} + \varepsilon)^3 \\ &= 6\varepsilon \cdot (\tfrac{1}{2} + \varepsilon) + 2 \cdot (\tfrac{1}{2} + \varepsilon) \cdot ((\tfrac{1}{2} - \varepsilon)^2 - (\tfrac{1}{2} + \varepsilon)^2) \\ &= 6\varepsilon \cdot (\tfrac{1}{2} + \varepsilon) + 2 \cdot (\tfrac{1}{2} + \varepsilon) \cdot (-2\varepsilon) = 2\varepsilon \cdot (\tfrac{1}{2} + \varepsilon) \end{aligned}$$

This means that one of the three coalitions can gain at least  $\frac{2}{3}\varepsilon(\frac{1}{2} + \varepsilon) = \frac{1}{3}\varepsilon + \frac{2}{3}\varepsilon^2$ . ◀

# Even the Easiest(?) Graph Coloring Problem Is Not Easy in Streaming!

**Anup Bhattacharya**

Indian Statistical Institute, Kolkata, India  
bhattacharya.anup@gmail.com

**Arijit Bishnu**

Indian Statistical Institute, Kolkata, India  
arijit@isical.ac.in

**Gopinath Mishra**

Indian Statistical Institute, Kolkata, India  
gopianjan117@gmail.com

**Anannya Upasana**

Indian Statistical Institute, Kolkata, India  
anannya.upasana23@gmail.com

---

## Abstract

We study a graph coloring problem that is otherwise easy in the RAM model but becomes quite non-trivial in the one-pass streaming model. In contrast to previous graph coloring problems in streaming that try to find an assignment of colors to vertices, our main work is on estimating the number of conflicting or monochromatic edges given a coloring function that is streaming along with the graph; we call the problem CONFLICT-EST. The coloring function on a vertex can be read or accessed only when the vertex is revealed in the stream. If we need the color on a vertex that has streamed past, then that color, along with its vertex, has to be stored explicitly. We provide algorithms for a graph that is streaming in different variants of the vertex arrival in one-pass streaming model, viz. the VERTEX ARRIVAL (VA), Vertex Arrival With Degree Oracle (VADEG), VERTEX ARRIVAL IN RANDOM ORDER (VARAND) models, with special focus on the random order model. We also provide matching lower bounds for most of the cases. The mainstay of our work is in showing that the properties of a random order stream can be exploited to design efficient streaming algorithms for estimating the number of monochromatic edges. We have also obtained a lower bound, though not matching the upper bound, for the random order model. Among all the three models vis-a-vis this problem, we can show a clear separation of power in favor of the VARAND model.

**2012 ACM Subject Classification** Theory of computation → Streaming, sublinear and near linear time algorithms; Mathematics of computing → Probabilistic algorithms

**Keywords and phrases** Streaming, random ordering, graph coloring, estimation, lower bounds

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.15

**Related Version** A full version of the paper is available at <https://arxiv.org/pdf/2010.13143.pdf>.

**Funding** Anup Bhattacharya: Funded by NPDF fellowship at ISI, Kolkata.

## 1 Introduction

The *chromatic number*  $\chi(G)$  of an  $n$ -vertex graph  $G = (V, E)$  is the minimum number of colors needed to color the vertices of  $V$  so that no two adjacent vertices get the same color. The *chromatic number* problem is NP-hard and even hard to approximate within a factor of  $n^{1-\varepsilon}$  for any constant  $\varepsilon > 0$  [14, 28, 20]. For any connected undirected graph  $G$  with maximum degree  $\Delta$ ,  $\chi(G)$  is at most  $\Delta + 1$  [27]. This existential coloring scheme can be made constructive across different models of computation. A seminal result of recent vintage is that the  $\Delta + 1$  coloring can be done in the streaming model [3]. Of late, there has been interest

in graph coloring problems in the sub-linear regime across a variety of models [1, 3, 4, 8, 6]. Keeping with the trend of coloring problems, these works look at assigning colors to vertices. Since the size of the output will be as large as the number of vertices, researchers study the semi-streaming model [22] for streaming graphs. In the semi-streaming model,  $\tilde{O}(n)^1$  space is allowed.

In a marked departure from the above works that look at the classical coloring problem, the starting point of our work is (inarguably?) the simplest question one can ask in graph coloring – given a coloring function  $f : V \rightarrow \{1, \dots, C\}$  on the vertex set  $V$  of a graph  $G = (V, E)$ , is  $f$  a valid coloring, i.e., for any edge  $e \in E$ , do both the endpoints of  $e$  have different colors? This is the problem one encounters while proving that the problem of chromatic number belongs to the class NP [15]. CONFLICT-EST, the problem of estimating the number of monochromatic (or, conflicting) edges for a graph  $G$  given a coloring function  $f$ , remains a simple problem in the RAM model; it even remains simple in the one-pass streaming model if the coloring function  $f$  is marked on a *public board*, readable at all times. We show that the problem throws up interesting consequences if the coloring function  $f$  on a vertex is revealed only when the vertex is revealed in the stream. For a streaming graph, if the vertices are assigned colors arbitrarily or randomly on-the-fly while it is exposed, our results can also be used to estimate the number of conflicting edges. These problems also find their use in estimating the number of conflicts in a job schedule and verifying a given job schedule in a streaming setting. This can also be extended to problems in various domains like frequency assignment in wireless mobile networks and register allocation [13]. As the problem, by its nature, admits an estimate or a yes-no answer, we can try for space efficient algorithms in the conventional graph streaming models like VERTEX ARRIVAL [11]. We also note in passing that many of the trend setting problems in streaming, like frequency moments, distinct elements, majority, etc. have been simple problems in the ubiquitous RAM model as the coloring problem we solve here.

## 2 Preliminaries

### 2.1 Notations and the streaming models

**Notations.** We denote the set  $\{1, \dots, n\}$  by  $[n]$ .  $G(V(G), E(G))$  denotes a graph where  $V(G)$  and  $E(G)$  denote the set of vertices and edges of  $G$ , respectively;  $|V| = n$  and  $|E| = m$ . We will write only  $V$  and  $E$  for vertices and edges when the graph is clear from the context. We denote  $E_M \subseteq E$  as the set of monochromatic edges. The set of neighbors of a vertex  $u \in V(G)$  is denoted by  $N_G(u)$  and the degree of a vertex  $u \in V(G)$  is denoted by  $d_G(u)$ . Let  $N_G(u) = N_G^-(u) \uplus N_G^+(u)$  where  $N_G^-(u)$  and  $N_G^+(u)$  denote the set of neighbors of  $u$  that have been exposed already and are yet to be exposed, respectively in the stream. Also,  $d_G(u) = d_G^-(u) + d_G^+(u)$  where  $d_G^-(u) = |N_G^-(u)|$  and  $d_G^+(u) = |N_G^+(u)|$ . For a monochromatic edge  $(u, v) \in E_M$ , we refer to  $u$  and  $v$  as monochromatic neighbors of each other. We define  $d_M(u)$  to be the number of monochromatic neighbors of  $u$  and hence, the monochromatic degree of  $u$ .

Let  $\mathbb{E}[X]$  denote the expectation of the random variable  $X$ . For an event  $\mathcal{E}$ ,  $\bar{\mathcal{E}}$  denotes the complement of  $\mathcal{E}$ .  $\mathbb{P}(\mathcal{E})$  denotes the probability of an event  $\mathcal{E}$ . The statement “event  $\mathcal{E}$  occurs with high probability” is equivalent to  $\mathbb{P}(\mathcal{E}) \geq 1 - \frac{1}{n^c}$ , where  $c$  is an absolute constant. The statement “ $a$  is a  $1 \pm \varepsilon$  multiplicative approximation of  $b$ ” means  $|b - a| \leq \varepsilon \cdot b$ . For  $x \in \mathbb{R}$ ,  $\exp(x)$  denotes the standard exponential function, that is,  $e^x$ . By polylogarithmic, we mean  $\mathcal{O}((\log n / \varepsilon)^{\mathcal{O}(1)})$ . The notation  $\tilde{O}(\cdot)$  hides a polylogarithmic term in  $\mathcal{O}(\cdot)$ .

---

<sup>1</sup>  $\tilde{O}(\cdot)$  hides a polylogarithmic factor.

**Streaming models for graphs.** As alluded to earlier, the crux of the problem depends on the way the coloring function  $f$  is revealed in the stream. The details follow.

- (i) **VERTEX ARRIVAL (VA):** The vertices of  $V$  are exposed in an arbitrary order. After a vertex  $v \in V$  is exposed, all the edges between  $v$  and pre-exposed neighbors of  $v$ , are revealed. This set of edges are revealed one by one in an arbitrary order. Along with the vertex  $v$ , only the color  $f(v)$  is exposed, and not the colors of any pre-exposed vertices. So, we can check the monochromaticity of an edge  $(v, u)$  only if  $u$  and  $f(u)$  are explicitly stored.
- (ii) **VERTEX ARRIVAL WITH DEGREE ORACLE (VADEG)** [23, 7]: This model works same as the VA model in terms of exposure of the vertex  $v$  and the coloring on it; but we are allowed to know the degree  $d_G(v)$  of the currently exposed vertex  $v$  from a degree oracle on  $G$ .
- (iii) **VERTEX ARRIVAL IN RANDOM ORDER (VARAND)** [25, 26]: This model works same as the VA model but the vertex sequence revealed is equally likely to be any one of the permutations of the vertices.
- (iv) **EDGE ARRIVAL (EA):** The stream consists of edges of  $G$  in an arbitrary order. As the edge  $e$  is revealed, so are the colors on its endpoints. Thus the conflicts can be easily checked.
- (v) **ADJACENCY LIST (AL):** The vertices of  $V$  are exposed in an arbitrary order. When a vertex  $v$  is exposed, all the edges that are incident to  $v$ , are revealed one by one in an arbitrary order. Note that in this model each edge is exposed twice, once for each exposure of an incident vertex. As in the VA model, here also only  $v$ 's color  $f(v)$  is exposed.

As the conflicts can be checked easily in the EA model in  $O(1)$  space, a logarithmic counter is enough to count the number of monochromatic edges. The AL model works almost the same as the VADEG model. So, we focus on the three models – VA, VADEG and VARAND in this work and show that they have a clear separation in their power vis-a-vis the problem we solve. A crucial takeaway from our work is that the random order assumption on exposure of vertices has huge improvements in space complexity.

## 2.2 Problem definitions, results and the ideas

**Problem definition.** Let the vertices of  $G$  be colored with a function  $f : V(G) \rightarrow [C]$ , for  $C \in \mathbb{N}$ . An edge  $(u, v) \in E(G)$  is said to be *monochromatic* or *conflicting* with respect to  $f$  if  $f(u) = f(v)$ . A coloring function  $f$  is called *valid* if no edge in  $E(G)$  is monochromatic with respect to  $f$ . For a given parameter  $\varepsilon \in (0, 1)$ ,  $f$  is said to be  $\varepsilon$ -far from being *valid* if at least  $\varepsilon \cdot |E(G)|$  edges are monochromatic with respect to  $f$ . We study the following problems.

► **Problem 2.1 (CONFLICT ESTIMATION aka CONFLICT-EST).** A graph  $G = (V, E)$  and a coloring function  $f : V(G) \rightarrow [C]$  are streaming inputs. Given an input parameter  $\varepsilon > 0$ , the objective is to estimate the number of monochromatic edges in  $G$  within a  $(1 \pm \varepsilon)$ -factor.

► **Problem 2.2 (CONFLICT SEPARATION aka CONFLICT-SEP).** A graph  $G = (V, E)$  and a coloring function  $f : V(G) \rightarrow [C]$  are streaming inputs. Given an input parameter  $\varepsilon > 0$ , the objective is to distinguish if the coloring function  $f$  is valid or is  $\varepsilon$ -far from being valid.

► Remark 2.3. Problem 2.1 is our main focus, but we will mention a result on Problem 2.2 in Section 5. Notice that the problem CONFLICT-EST is at least as hard as CONFLICT-SEP.

**The results and the ideas involved.** All our upper and lower bounds on space are for one-pass streaming algorithms. Table 1 states our results for the CONFLICT-EST problem, the main problem we solve in this paper, across different variants of the VA model. The main thrust of our work is on estimating monochromatic edges under random order stream. For random order stream, we present both upper and lower bounds in Sections 3 and 4. There is a gap between the upper and lower bounds in the VARAND model, though we have a strong hunch that our upper bound is tight. Apart from the above, using a structural result on graphs, we show in Section 5 that the CONFLICT-SEP problem admits an easy algorithm in the VARAND model. To give a complete picture across different variants of VA models, we show matching upper and lower bounds for constant  $\varepsilon > 0$  in the VA and VADEG models in [9]<sup>2</sup>.

■ **Table 1** This table shows our results on CONFLICT-EST on a graph  $G(V, E)$  across different VERTEX ARRIVAL models. Here,  $T > 0$  denotes the promised lower bound on the number of monochromatic edges. This paper discusses the results mentioned in the middle column corresponding to VARAND. The other results are discussed in the full version of the paper [9].

Model	VA	VARAND	VADEG
Upper Bound	$\tilde{O}\left(\min\{ V , \frac{ V ^2}{T}\}\right)$ (Thm. 3.1 in [9])	$\tilde{O}\left(\frac{ V }{\sqrt{T}}\right)$ (Sec. 3, Thm. 3.1)	$\tilde{O}\left(\min\{ V , \frac{ E }{T}\}\right)$ (Thm. 3.2 in [9])
Lower Bound	$\Omega\left(\min\{ V , \frac{ V ^2}{T}\}\right)$ (Thm. E.1 in [9])	$\Omega\left(\frac{ V }{T^2}\right)$ (Sec. 4, Thm. 4.1)	$\Omega\left(\min\{ V , \frac{ E }{T}\}\right)$ (Thm. E.2 in [9])

The promise  $T$  on the number of monochromatic edges is a very standard assumption for estimating substructures in the world of graph streaming algorithm [17, 19, 18, 23, 5].<sup>3</sup>

We now briefly mention the salient ideas involved. For the simpler variant of CONFLICT-EST in VA model, we first check if  $|V| \geq T$ . If yes, we store all the vertices and their colors in the stream to determine the exact value of the number of monochromatic edges. Otherwise, we sample each pair of vertices  $\{u, v\}$  in  $\binom{V}{2}$ <sup>4</sup>, with probability  $\tilde{O}(1/T)$  independently<sup>5</sup> before the stream starts. When the stream comes, we compute the number of monochromatic edges from this sample. The details are in Section 3 of [9]. Though the algorithm looks extremely simple, it matches the lower bound result for CONFLICT-EST in VA model, presented in Appendix E of [9]. The VADEG model with its added power of a degree oracle, allows us to know  $d_G(u)$  for a vertex  $u$  and as edges to pre-exposed vertices are revealed, we also know  $d_G^-(u)$  and  $d_G^+(u)$ . This allows us to use sampling to store vertices and to use a technique which we call *sampling into the future* where indices of random neighbors, out of  $d_G^+(u)$  neighbors, are selected for future checking. The upper bound result for CONFLICT-EST in VADEG model, presented in Section 3 of [9], is tight as we also prove a matching lower bound in Appendix E of [9].

<sup>2</sup> The reference [9] is the full version of this submission.

<sup>3</sup> Here we have cited a few. However, there are huge amount of relevant literature.

<sup>4</sup>  $\binom{V}{2}$  denotes the set of all size 2 subsets of  $V(G)$ .

<sup>5</sup> Note that we might sample some pairs that are not forming edges in the graph.



The algorithm for CONFLICT-EST in VARAND model is the mainstay of our work and is presented in Section 3. We redefine the degree in terms of the number of monochromatic neighbors a vertex has in the randomly sampled set. Here, we estimate the high monochromatic degree and low monochromatic degree vertices separately by sampling a random subset of vertices. While the monochromatic degree for the high degree vertices can be extrapolated from the sample, handling low monochromatic degree vertices individually in the same way does not work. To get around, we group such vertices having similar monochromatic degree and treat them as an entity. We also provide a lower bound for the VARAND model, in Section 4, using a reduction from *multi-party set disjointness*; though there is a gap in terms of the exponent in  $T$ .

The highlights of our work are as follows:

- We show that possibly the easiest graph coloring problem is worth studying over streams.
- For researchers working in streaming, the *gold standard* is the EA model as most problems are non-trivial in this model. We point out a problem that is harder to solve in the VA model as compared to the EA model.
- We show that the three VA related models have a clear separation in their space complexities vis-a-vis the problem we solve. We could exploit the random order of the arrival of the vertices to get substantial improvements in space complexity.
- We could obtain lower bounds for all the three models and the lower bounds are matching for the VA and VADEG models.

### 2.3 Prior works on graph coloring in semi-streaming model.

Bera and Ghosh [8] commenced the study of vertex coloring in the semi-streaming model. They devise a randomized one pass streaming algorithm that finds a  $(1 + \varepsilon)\Delta$  vertex coloring in  $\tilde{O}(n)$  space. Assadi et al. [3] find a proper vertex coloring using  $\Delta + 1$  colors via various classes of sublinear algorithms. Their state of the art contributions can be attributed to a key result called the *palette-sparsification theorem* which states that for an  $n$ -vertex graph with maximum degree  $\Delta$ , if  $O(\log n)$  colors are sampled independently and uniformly at random for each vertex from a list of  $\Delta + 1$  colors, then with a high probability a proper  $\Delta + 1$  coloring exists for the graph. They design a randomized one-pass dynamic streaming algorithm for the  $\Delta + 1$  coloring using  $\tilde{O}(n)$  space. The algorithm takes post-processing  $\tilde{O}(n\sqrt{\Delta})$  time and assumes a prior knowledge of  $\Delta$ . Alon and Assadi [2] improve the palette sparsification result of [3]. They consider situations where the number of colors available is both more than and less than  $\Delta + 1$  colors. They show that sampling  $O_\varepsilon(\sqrt{\log n})$  colors per vertex is sufficient and necessary for a  $(1 + \varepsilon)\Delta$  coloring. Bera et al. [6] give a new graph coloring algorithm in the semi-streaming model where the number of colors used is parameterized by the degeneracy  $\kappa$ . The key idea is a *low degeneracy partition*, also employed in [8]. The numbers of colors used to properly color the graph is  $\kappa + o(\kappa)$  and post-processing time of the algorithm is improved to  $\tilde{O}(n)$ , without any prior knowledge about  $\kappa$ . Behnezhad et al. [4] were the first to give one-pass W-streaming algorithms (streaming algorithms where outputs are produced in a streaming fashion as opposed to outputs given finally at the end) for edge coloring both when the edges arrive in a random order or in an adversarial fashion.

### 3 CONFLICT-EST in VARAND model

In this Section, we show that the power of randomness can be used to design a better solution for the CONFLICT-EST problem in the VARAND model. The CONFLICT-EST problem is the main highlight of our work. We feel that the crucial use of randomness in the input that is used to estimate a substructure (here, monochromatic edges) in a graph, will be of independent interest.

In this variant, we are given an  $\varepsilon \in (0, 1)$  and a promised lower bound  $T$  on  $|E_M|$ , the number of monochromatic edges in  $G$ , as input and our objective is to determine a  $(1 \pm \varepsilon)$ -approximation to  $|E_M|$ .

► **Theorem 3.1.** *Given any graph  $G = (V, E)$  and a coloring function  $f : V(G) \rightarrow [C]$  as input in the stream, the CONFLICT-EST problem in the VARAND model can be solved with high probability in  $\tilde{O}\left(\frac{|V|}{\sqrt{T}}\right)$  space, where  $T$  is a lower bound on the number of monochromatic edges in the graph.*

#### The proof idea

##### A random sample comes for free – pick the first few vertices

Let  $v_1, \dots, v_n$  be the random ordering in which the vertices of  $V$  are revealed. Let  $R$  be a random subset of  $\Gamma = \tilde{O}\left(\frac{n}{\sqrt{T}}\right)$  many vertices of  $G$  sampled without replacement<sup>6</sup>. As we are dealing with a random order stream, consider the first  $\Gamma$  vertices in the stream; they can be treated as  $R$ , the random sample. We start by storing all the vertices in  $R$  as well as their colors. Observe that if the monochromatic degree of any vertex  $v_i$  is *large* (say roughly more than  $\sqrt{T}$ ), then it can be well approximated by looking at the number of monochromatic neighbors that  $v_i$  has in  $R$ . As a vertex  $v_i$  streams past, there is no way we can figure out its monochromatic degree, unless we store its monochromatic neighbors that appear before it in the stream; if we could, we were done. Our only savior is the stored random subset  $R$ .

##### Classifying the vertices of the random sample $R$ based on its monochromatic degree

Our algorithm proceeds by figuring out the influence of the color of  $v_i$  on the monochromatic degrees of vertices in  $R$ . To estimate this, let  $\kappa_{v_i}$  denote the number of monochromatic neighbors that  $v_i$  has in  $R$ . We set a threshold  $\tau = \frac{|R|}{n} \frac{\sqrt{\varepsilon T}}{8t}$ , where  $t = \lceil \log_{1+\frac{\varepsilon}{10}} n \rceil$ . The significance of  $t$  will be clear from the discussion below. Any vertex  $v_i$  will be classified as a high- $m_R$  or low- $m_R$  degree vertex depending on its monochromatic degree within  $R$ , i.e., if  $\kappa_{v_i} \geq \tau$ , then  $v_i$  is a high- $m_R$  vertex, else it is a low- $m_R$  vertex, respectively. (We use the subscripts  $m_R$  to stress the fact that the monochromatic degrees are induced by the set  $R$ .) Let  $H$  and  $L$  be the partition of  $V$  into the set of high- $m_R$  and low- $m_R$  degree vertices in  $G$ . Let  $H_R$  and  $L_R$  denote the set of high- $m_R$  and low- $m_R$  degree vertices in  $R$ . Notice that, because of the definition of high- $m_R$  and low- $m_R$  degree vertices, not only the sets  $H_R, L_R$  are subsets of  $R$ , but they are determined by the vertices of  $R$  only.

Let  $m_h$  and  $m_\ell$  denote the sum of the monochromatic degrees of all the high- $m_R$  degree vertices and low- $m_R$  degree vertices in  $G$ , respectively. So,  $m_h = \sum_{v \in H} d_M(v)$  and  $m_\ell = \sum_{v \in L} d_M(v)$ . Note that  $\hat{m} = |E_M| = \frac{1}{2} \sum_{v \in V} d_M(v) = \frac{1}{2} (m_h + m_\ell)$ . We will describe how to approximate  $m_h$  and  $m_\ell$  separately. The formal algorithm is described in Algorithm 1

<sup>6</sup>  $\tilde{O}(\cdot)$  hides a polynomial factor of  $\log n$  and  $\frac{1}{\varepsilon}$  in the upper bound.

as  $\text{RANDOM-ORDER-EST}(\varepsilon, T)$  (in Appendix D) that basically executes steps to approximate  $m_h$  and  $m_\ell$  in parallel.

### To approximate $m_h$ , the random sample $R$ comes to rescue

We can find  $\widehat{m}_h$ , that is, a  $(1 \pm \frac{\varepsilon}{10})$  approximation of  $m_h$  as described below. For each vertex  $v_i \in R$  and each monochromatic edge  $(u, v_i)$ ,  $u \in R$ , we see in the stream, we increase the value of  $\kappa_u$  for  $u$  and  $\kappa_{v_i}$  for  $v_i$ . After all the vertices in  $R$  are revealed, we can determine  $H_R$  by checking whether  $\kappa_{v_i} \geq \tau$  for each  $v_i \in R$ . For each vertex  $v_i \in H_R$ , we set its approximate monochromatic degree  $\widehat{d}_{v_i}$  to be  $\frac{n}{|R|}\kappa_{v_i}$ . We initialize the estimated sum of the monochromatic degree of high degree vertices as  $\widehat{m}_h = \sum_{v_i \in H_R} \widehat{d}_{v_i}$ . For each vertex  $v_i \notin R$  in the stream, we can determine  $\kappa_{v_i}$ , as we have stored all the vertices in  $R$  along with their colors, and hence we can also determine whether  $v_i$  is a high- $m_R$  degree vertex in  $G$ . If  $v_i \notin R$  is a high- $m_R$  degree vertex, we determine  $\widehat{d}_{v_i} = \frac{n}{|R|}\kappa_{v_i}$  and update  $\widehat{m}_h$  by  $\widehat{m}_h + \widehat{d}_{v_i}$ . Observe that, at the end,  $\widehat{m}_h$  is  $\sum_{v_i \in H} \widehat{d}_{v_i}$ . Recall that  $H$  is the set of all high- $m_R$  degree vertices in  $G$ . For each  $v_i \in H$ , we will show, as in Claim 3.3, that  $\widehat{d}_{v_i}$  is a  $(1 \pm \frac{\varepsilon}{10})$ -approximation to  $d_M(v_i)$  with high probability. This implies that

$$\left(1 - \frac{\varepsilon}{10}\right) m_h \leq \widehat{m}_h \leq \left(1 + \frac{\varepsilon}{10}\right) m_h \quad (1)$$

### To approximate $m_\ell$ , group the vertices in $L$ based on similar monochromatic degree

Recall that  $m_\ell = \sum_{v_i \in L} d_M(v_i)$ . Unlike the high- $m_R$  degree vertices, it is not possible to approximate the monochromatic degree of  $v_i \in L$  from  $\kappa_{v_i}$ . To cope up with this problem, we partition the vertices of  $L$  into  $t$  buckets  $B_1, \dots, B_t$  such that all the vertices present in a bucket have *similar* monochromatic degrees, where  $t = \lceil \log_{1+\frac{\varepsilon}{10}} n \rceil$ . The bucket  $B_j$  is defined as follows:  $B_j = \{v_i \in L : (1 + \frac{\varepsilon}{10})^{j-1} \leq d_M(v_i) < (1 + \frac{\varepsilon}{10})^j\}$ .

Note that our algorithm will not find the buckets explicitly. It will be used for the analysis only. Observe that  $\sum_{j \in [t]} |B_j| (1 + \frac{\varepsilon}{10})^{j-1} \leq m_\ell < \sum_{j \in [t]} |B_j| (1 + \frac{\varepsilon}{10})^j$ . We can surely approximate  $m_\ell$  by approximating  $|B_j|$ s suitably. We estimate  $|B_j|$ s as follows. After the stream of the vertices in  $R$  has gone past, we have the set of low- $m_R$  degree vertices  $L_R$  in  $R$  and  $\widehat{d}_{v_i} = \kappa_{v_i}$  for each  $v_i \in L_R$ . For each  $v_i \notin R$  in the stream, we determine the monochromatic neighbors of  $v_i$  in  $L_R$ . It is possible as we have stored all the vertices in  $R$  and their colors. For each monochromatic neighbor  $v_{i'} \in L_R$  of  $v_i$ , we increase the value of  $\widehat{d}_{v_{i'}}$  of  $v_{i'}$ . Observe that, at the end of the stream,  $\widehat{d}_{v_{i'}} = d_M(v_{i'})$  for each  $v_{i'} \in L_R$ , i.e., we can accurately estimate the monochromatic degree of each  $v_{i'} \in L_R$ . So, we can determine the bucket where each vertex in  $L_R$  belongs. Let  $A_j (= L_R \cap B_j)$  be the bucket  $B_j$  projected onto  $L_R$  in the random sample; note that as  $B_j \subseteq L$  and  $L_R = L \cap R$ ,  $A_j = R \cap B_j$  also. We determine  $\widehat{m}_\ell = \frac{n}{|R|} \sum_{j \in [t]} |A_j| (1 + \frac{\varepsilon}{10})^j$ . We can show that  $\frac{n}{|R|} |A_j|$  is a  $(1 + \frac{\varepsilon}{10})$ -approximation of  $|B_j|$ , with high probability, if  $|B_j| \geq \frac{\sqrt{\varepsilon T}}{10t}$ . Also, we can show that, if  $|B_j| < \frac{\sqrt{\varepsilon T}}{10t}$ , then  $|A_j| \leq \frac{|R|}{n} \frac{\sqrt{\varepsilon T}}{8t}$  with high probability. Now using the fact that we consider bucketing of only low- $m_R$  degree vertices ( $L_R$ ), we can show that

$$\left(1 - \frac{\varepsilon}{10}\right) \left(m_\ell - \frac{\varepsilon T}{63t}\right) \leq \widehat{m}_\ell \leq \left(1 + \frac{\varepsilon}{10}\right)^2 \left(m_\ell + \frac{\varepsilon T}{56t}\right). \quad (2)$$

Note that  $\varepsilon \in (0, 1)$  and  $t = \lceil \log_{1+\frac{\varepsilon}{10}} n \rceil$ . Assuming  $T \geq 63t^2$ , Equations 1 and 2 imply that  $\widehat{m} = \frac{1}{2}(\widehat{m}_h + \widehat{m}_\ell)$  is a  $(1 \pm \varepsilon)$ -approximation to  $|E_M|$ . If  $T < 63t^2$ , then note that

## 15:8 Even the Easiest(?) Graph Coloring Problem Is Not Easy in Streaming!

$n = \tilde{O}\left(\frac{n}{\sqrt{T}}\right)$ . So, in that case, we store all the vertices along with their colors and compute the exact value of  $|E_M|$ .

### Proof of correctness

The correctness of the algorithm follows trivially if  $T < 63t^2$ . So, let us assume that  $T \geq 63t^2$ . In the VARAND model, we consider the first  $\tilde{\Theta}\left(\frac{n}{\sqrt{T}}\right)$  vertices as the random sample  $R$  without replacement. Using the Chernoff bound for sampling without replacement (See Lemma A.2 in Appendix A), we can have the following lemma (The proof is in Appendix B), which will be useful for the correctness proof of Algorithm 1 (RANDOM-ORDER-EST( $\varepsilon, T$ )) in case of  $T \geq 63t^2$ .

#### ► Lemma 3.2.

- (i) For each  $j \in [t]$  with  $|B_j| \geq \frac{\sqrt{\varepsilon T}}{10t}$ ,  $\mathbb{P}\left(\left|B_j \cap R\right| - \frac{|R||B_j|}{n} \geq \frac{\varepsilon}{10} \frac{|R||B_j|}{n}\right) \leq \frac{1}{n^{10}}.$
- (ii) For each  $j \in [t]$  with  $|B_j| < \frac{\sqrt{\varepsilon T}}{10t}$ ,  $\mathbb{P}\left(|B_j \cap R| \geq \frac{|R|}{n} \frac{\sqrt{\varepsilon T}}{8t}\right) \leq \frac{1}{n^{10}}.$
- (iii) For each vertex  $v_i$  with  $d_M(v_i) \geq \frac{\sqrt{\varepsilon T}}{10t}$ ,  $\mathbb{P}\left(\left|\kappa_{v_i} - \frac{|R|d_M(v_i)}{n}\right| \geq \frac{\varepsilon}{10} \frac{|R|d_M(v_i)}{n}\right) \leq \frac{1}{n^{10}}.$
- (iv) For each vertex  $v_i$  with  $d_M(v_i) < \frac{\sqrt{\varepsilon T}}{10t}$ ,  $\mathbb{P}\left(\kappa_{v_i} \geq \frac{|R|}{n} \frac{\sqrt{\varepsilon T}}{8t}\right) \leq \frac{1}{n^{10}}.$

The correctness proof of the algorithm is divided into the following two claims.

▷ Claim 3.3.  $(1 - \frac{\varepsilon}{10})m_h \leq \widehat{m}_h \leq (1 + \frac{\varepsilon}{10})m_h$  with probability at least  $1 - \frac{1}{n^9}.$

▷ Claim 3.4.  $(1 - \frac{\varepsilon}{10})(m_\ell - \frac{\varepsilon T}{63t}) \leq \widehat{m}_\ell \leq (1 + \frac{\varepsilon}{10})^2(m_\ell + \frac{\varepsilon T}{56t})$  with probability at least  $1 - \frac{1}{n^7}.$

Assuming the above two claims hold and taking  $\varepsilon \in (0, 1)$ ,  $t = \lceil \log_{1+\frac{\varepsilon}{10}} n \rceil$  and  $T \geq 63t^2$ , observe that  $\widehat{m} = \frac{1}{2}(\widehat{m}_h + \widehat{m}_\ell)$  is a  $(1 \pm \varepsilon)$  approximation of  $|E_M| = m_h + m_\ell$  with high probability. Thus, it remains to prove Claims 3.3 and 3.4.

Proof of Claim 3.3. Note that  $m_h = \sum_{v_i: \kappa_{v_i} \geq \frac{|R|}{n} \frac{\sqrt{\varepsilon T}}{8t}} d_M(v_i)$  and  $\widehat{m}_h = \sum_{v_i: \kappa_{v_i} \geq \frac{|R|}{n} \frac{\sqrt{\varepsilon T}}{8t}} \widehat{d}_{v_i}.$

From Lemma 3.2 (iv) and (iii),  $\kappa_{v_i} \geq \frac{|R|}{n} \frac{\sqrt{\varepsilon T}}{8t}$  implies that  $\widehat{d}_{v_i}$  is an  $(1 \pm \frac{\varepsilon}{10})$  approximation to  $d_M(v_i)$  with probability at least  $1 - \frac{1}{n^{10}}$ . Hence, we have  $(1 - \frac{\varepsilon}{10})m_h \leq \widehat{m}_h \leq (1 + \frac{\varepsilon}{10})m_h$  with probability at least  $1 - \frac{1}{n^9}.$  ◁

Proof of Claim 3.4. Note that  $m_\ell = \sum_{v_i \in L} d_M(v_i) = \sum_{v_i: \kappa_{v_i} < \frac{|R|}{n} \frac{\sqrt{\varepsilon T}}{8t}} d_M(v_i)$  and

$\widehat{m}_\ell = \frac{n}{|R|} \sum_{j \in [t]} |A_j| (1 + \frac{\varepsilon}{10})^j.$  Recall that the vertices in  $L$  are partitioned into  $t$  buckets as follows:

$B_j = \{v_i \in L : (1 + \frac{\varepsilon}{10})^{j-1} \leq d_M(v_i) < (1 + \frac{\varepsilon}{10})^j\}$ , where  $j \in [t]$ . By Lemma 3.2 (iv),  $\kappa_{v_i} < \frac{|R|}{n} \frac{\sqrt{\varepsilon T}}{8t}$  implies that  $d_M(v_i) \leq \frac{\sqrt{\varepsilon T}}{7t}$  with probability  $1 - \frac{1}{n^{10}}$ . So, we have the following observation.

► **Observation 3.5.** Let  $j \in [t]$  be such that  $|A_j| \neq 0$  ( $|B_j| \neq 0$ ). Then, with probability at least  $1 - \frac{1}{n^{10}}$ , the monochromatic degree of each vertex in  $A_j$  as well as  $B_j$  is at most  $\frac{\sqrt{\varepsilon T}}{7t}$ , that is,  $(1 + \frac{\varepsilon}{10})^j \leq \frac{\sqrt{\varepsilon T}}{7t}.$

To upper and lower bound  $\widehat{m}_\ell$  in terms of  $m_\ell$ , we upper and lower bound  $m_\ell$  in terms of  $|B_j|$ 's as follows; for the upper bound, we break the sum into two parts corresponding to large and small sized buckets:

$$\begin{aligned} \sum_{j \in [t]} |B_j| \left(1 + \frac{\varepsilon}{10}\right)^{j-1} &\leq m_\ell < \sum_{j \in [t]} |B_j| \left(1 + \frac{\varepsilon}{10}\right)^j \\ \sum_{j \in [t]} |B_j| \left(1 + \frac{\varepsilon}{10}\right)^{j-1} &\leq m_\ell < \sum_{j \in [t]: |B_j| \geq \frac{\sqrt{\varepsilon T}}{9t}} |B_j| \left(1 + \frac{\varepsilon}{10}\right)^j + \sum_{j \in [t]: |B_j| < \frac{\sqrt{\varepsilon T}}{9t}} |B_j| \left(1 + \frac{\varepsilon}{10}\right)^j \end{aligned}$$

By Observation 3.5, we bound  $m_\ell$  in terms of  $|B_j|$ 's with probability  $1 - \frac{1}{n^9}$ .

$$\sum_{j \in [t]} |B_j| \left(1 + \frac{\varepsilon}{10}\right)^{j-1} \leq m_\ell < \sum_{j \in [t]: |B_j| \geq \frac{\sqrt{\varepsilon T}}{9t}} |B_j| \left(1 + \frac{\varepsilon}{10}\right)^j + t \cdot \frac{\sqrt{\varepsilon T}}{9t} \frac{\sqrt{\varepsilon T}}{7t}$$

This implies the following Observation:

► **Observation 3.6.**  $\sum_{j \in [t]} |B_j| \left(1 + \frac{\varepsilon}{10}\right)^{j-1} \leq m_\ell < \sum_{j \in [t]: |B_j| \geq \frac{\sqrt{\varepsilon T}}{9t}} |B_j| \left(1 + \frac{\varepsilon}{10}\right)^j + \frac{\varepsilon T}{63t}$  holds with probability at least  $1 - \frac{1}{n^9}$ .

Now, we have all the ingredients to show that  $\widehat{m}_\ell$  is a  $(1 \pm \varepsilon)$  approximation of  $m_\ell$ . To get to  $\widehat{m}_\ell$ , we need to focus on low- $m_R$  vertices of  $R$ , i.e.,  $A_j$ 's. Breaking  $\widehat{m}_\ell = \frac{n}{|R|} \sum_{j \in [t]} |A_j| \left(1 + \frac{\varepsilon}{10}\right)^j$  depending on small and large values of  $|A_j|$ 's (recall  $A_j = L_R \cap B_j = R \cap B_j$ ), we have

$$\widehat{m}_\ell = \frac{n}{|R|} \left[ \sum_{j \in [t]: |A_j| \geq \frac{|R|}{n} \frac{\sqrt{\varepsilon T}}{8t}} |A_j| \left(1 + \frac{\varepsilon}{10}\right)^j + \sum_{j \in [t]: |A_j| < \frac{|R|}{n} \frac{\sqrt{\varepsilon T}}{8t}} |A_j| \left(1 + \frac{\varepsilon}{10}\right)^j \right] \quad (3)$$

Note that  $A_j = B_j \cap R$ . By Lemma 3.2 (ii),  $|A_j| \geq \frac{|R|}{n} \frac{\sqrt{\varepsilon T}}{8t}$  implies  $|B_j| \geq \frac{\sqrt{\varepsilon T}}{10t}$  with probability at least  $1 - \frac{1}{n^{10}}$ . Also, applying Lemma 3.2 (i),  $|B_j| \geq \frac{\sqrt{\varepsilon T}}{10t}$  implies  $|A_j|$  is an  $(1 \pm \frac{\varepsilon}{10})$ -approximation to  $\frac{|R||B_j|}{n}$  with probability at least  $1 - \frac{1}{n^{10}}$ . So, we have the following observation.

► **Observation 3.7.** Let  $j \in [t]$  be such that  $|A_j| \geq \frac{|R|}{n} \frac{\sqrt{\varepsilon T}}{8t}$ . Then  $|A_j|$  is an  $(1 \pm \frac{\varepsilon}{10})$ -approximation to  $\frac{|R||B_j|}{n}$  with probability at least  $1 - \frac{2}{n^{10}}$ , that is,  $\frac{n}{|R|} |A_j|$  is an  $(1 \pm \frac{\varepsilon}{10})$ -approximation to  $|B_j|$  with probability at least  $1 - \frac{2}{n^{10}}$ .

By the above observation along with Equation 3, we have the following upper bound on  $\widehat{m}_\ell$  with probability at least  $1 - \frac{1}{n^9}$ .

$$\begin{aligned} \widehat{m}_\ell &\leq \sum_{j \in [t]: |A_j| \geq \frac{|R|}{n} \frac{\sqrt{\varepsilon T}}{8t}} \left(1 + \frac{\varepsilon}{10}\right) |B_j| \left(1 + \frac{\varepsilon}{10}\right)^j + \sum_{j \in [t]: |A_j| < \frac{|R|}{n} \frac{\sqrt{\varepsilon T}}{8t}} \frac{n}{|R|} |A_j| \left(1 + \frac{\varepsilon}{10}\right)^j \\ &\leq \left(1 + \frac{\varepsilon}{10}\right)^2 \left[ \sum_{j \in [t]: |A_j| \geq \frac{|R|}{n} \frac{\sqrt{\varepsilon T}}{8t}} |B_j| \left(1 + \frac{\varepsilon}{10}\right)^{j-1} + \sum_{j \in [t]: |A_j| < \frac{|R|}{n} \frac{\sqrt{\varepsilon T}}{8t}} \frac{\sqrt{\varepsilon T}}{8t} \left(1 + \frac{\varepsilon}{10}\right)^{j-2} \right] \end{aligned}$$

Now by Observations 3.6 and 3.5, we have the following with probability at least  $1 - \frac{1}{n^8}$ .

$$\begin{aligned} \widehat{m}_\ell &\leq \left(1 + \frac{\varepsilon}{10}\right)^2 \left( m_\ell + t \cdot \frac{\sqrt{\varepsilon T}}{8t} \frac{\sqrt{\varepsilon T}}{7t} \right) \\ &= \left(1 + \frac{\varepsilon}{10}\right)^2 \left( m_\ell + \frac{\varepsilon T}{56t} \right) \end{aligned}$$

## 15:10 Even the Easiest(?) Graph Coloring Problem Is Not Easy in Streaming!

Now, we will lower bound  $\widehat{m}_\ell$ . From Equation 3, we have

$$\widehat{m}_\ell \geq \frac{n}{|R|} \sum_{j \in [t]: |A_j| \geq \frac{|R|}{n} \frac{\sqrt{\varepsilon T}}{8t}} |A_j| \left(1 + \frac{\varepsilon}{10}\right)^j$$

By Observation 3.7,  $|A_j| \geq \frac{|R|}{n} \frac{\sqrt{\varepsilon T}}{8t}$  implies  $\frac{n}{|R|} |A_j|$  is an  $(1 \pm \frac{\varepsilon}{10})$ -approximation to  $|B_j|$  with probability at least  $1 - \frac{2}{n^{10}}$ . So, the following lower bound on  $\widehat{m}_\ell$  holds with probability at least  $1 - \frac{1}{n^9}$ .

$$\widehat{m}_\ell \geq \left(1 - \frac{\varepsilon}{10}\right) \sum_{j \in [t]: |A_j| \geq \frac{|R|}{n} \frac{\sqrt{\varepsilon T}}{8t}} |B_j| \left(1 + \frac{\varepsilon}{10}\right)^j$$

By Lemma 3.2 (i), if  $|B_j| \geq \frac{\sqrt{\varepsilon T}}{9t}$ , then  $|A_j| \geq \frac{\sqrt{\varepsilon T}}{8t}$  with probability at least  $1 - \frac{1}{n^{10}}$ . Hence, we have the following lower bound on  $\widehat{m}_\ell$  with probability at least  $1 - \frac{1}{n^8}$ .

$$\widehat{m}_\ell \geq \left(1 - \frac{\varepsilon}{10}\right) \sum_{j \in [t]: |B_j| \geq \frac{\sqrt{\varepsilon T}}{9t}} |B_j| \left(1 + \frac{\varepsilon}{10}\right)^j$$

Now by Observation 3.6, we have the following with probability at least  $1 - \frac{1}{n^7}$ .

$$\widehat{m}_\ell \geq \left(1 - \frac{\varepsilon}{10}\right) \left(m_\ell - \frac{\varepsilon T}{63t}\right). \quad \triangleleft$$

### 4 Lower bound for CONFLICT-EST in VARAND model

In this Section, we show a lower bound of  $\Omega\left(\frac{n}{T^2}\right)$  for CONFLICT-EST in VERTEX ARRIVAL IN RANDOM ORDER via a reduction from a variation of MULTIPARTY SET DISJOINTNESS problem called DISJOINTNESS<sub>R</sub>( $t, n, p$ ), played among  $p$  players: Consider a matrix of order  $t \times n$  having  $t$  (rows) vectors  $M_1, \dots, M_t \in \{0, 1\}^n$  such that each entry of matrix  $M$  is given to one of the  $p$  players chosen uniformly at random. The objective is to determine whether there exists a column where all the entries are 1s. If  $t \geq 2$  and  $p = \Omega(t^2)$ , Chakrabarti et al. showed that any randomized protocol requires  $\Omega\left(\frac{n}{t}\right)$  bits of communication [10]. They showed that the lower bound holds under a promise called the UNIQUE INTERSECTION PROMISE which states that there exists at most a single column where all the entries are 1s and every other column of the matrix has Hamming weight either 0 or 1. Moreover, the lower bound holds even if all the  $p$  players know the random partition of the entries of matrix  $M$ .

► **Theorem 4.1.** *Let  $n, T \in \mathbb{N}$  be such that  $4 \leq T \leq \binom{n}{2}$ . Any constant pass streaming algorithm that takes the vertices and edges of a graph  $G(V, E)$  (with  $|V| = \Theta(n)$  and  $|E| = \Theta(m)$ ) and a coloring function  $f: V \rightarrow [C]$  in the VARAND model, and determines whether the monochromatic edges in  $G$  is 0 or  $\Omega(T)$  with probability  $2/3$ , requires  $\Omega\left(\frac{n}{T^2}\right)$  bits of space.*

**Proof.** Without loss of generality, assume that  $\sqrt{T} \in \mathbb{N}$ . Consider the DISJOINTNESS<sub>R</sub> $\left(\sqrt{T}, \frac{n}{\sqrt{T}}, p\right)$  problem with UNIQUE INTERSECTION PROMISE when all of the  $p$  players know the random partition of the entries of the relevant matrix  $M$ . Note that  $M$  is of order  $\left[\sqrt{T}\right] \times \left[\frac{n}{\sqrt{T}}\right]$  and  $p = AT$  for some suitable constant  $A \in \mathbb{N}$ . Also, consider a graph  $G$ , with  $V(G) = \{v_{ij} : i \in [\sqrt{T}], j \in [\frac{n}{\sqrt{T}}]\}$ , having  $\frac{n}{\sqrt{T}}$  many vertex disjoint cliques such that  $\{v_{1j}, \dots, v_{\sqrt{T}j}\}$  forms a clique for each  $j \in [n]$ , i.e., a column of  $M$  forms a clique. Also, notice

that each clique has  $\Theta(T)$  edges. Let us assume that there is an  $r$ -pass streaming algorithm  $\mathcal{S}$ , with space complexity  $s$  bits, that solves CONFLICT-EST for the above graph  $G$  in the VARAND model. Now, we give a protocol  $\mathcal{A}$  for  $\text{DISJOINTNESS}_R\left(\sqrt{T}, \frac{n}{\sqrt{T}}, p\right)$  with communication cost  $O(rsp)$ . Using the fact that the lower bound of  $\text{DISJOINTNESS}\left(\sqrt{T}, \frac{n}{\sqrt{T}}, p\right)$  is  $\Omega\left(\frac{n/\sqrt{T}}{\sqrt{T}}\right)$  along with the fact that  $p = AT$  and  $r$  is a constant, we get  $s = \Omega\left(\frac{n}{T^2}\right)$ .

### Protocol $\mathcal{A}$ for $\text{DISJOINTNESS}_R\left(\sqrt{T}, \frac{n}{\sqrt{T}}, p\right)$

Let  $P_1, \dots, P_p$  denote the set of  $p$  players. For  $k \in [p]$ ,  $V_k = \{v_{ij} : M_{ij} \text{ is with } P_k\}$ , where  $M_{ij}$  denotes the element present in the  $i$ -th row and  $j$ -th column of matrix  $M$ . Note that there is a one-to-one correspondence between the entries of  $M$  and the vertices in  $V(G)$ . Furthermore, there is a one-to-one correspondence between the columns of matrix  $M$  and the cliques in graph  $G$ . We assume that all the  $p$  players know the graph structure completely as well as both the one-to-one correspondences. The protocol proceeds as follows: for each  $k \in [p]$ , player  $P_k$  determines a random permutation  $\pi_k$  of the vertices in  $V_k$ . Also, for each  $k \in [p]$ , player  $P_k$  determines the colors of the vertices in  $V_k$  by the following rule: if  $M_{ij} = 1$ , then color vertex  $v_{ij}$  with color  $C_*$ . Otherwise, for  $M_{ij} = 0$ , color vertex  $v_{ij}$  with color  $C_i$ . Player  $P_1$  initiates the streaming algorithm and it goes over  $r$ -rounds.

**Rounds 1 to  $r - 1$ :** For  $k \in [p]$ , each player resumes the streaming algorithm by exposing the vertices in  $V_k$ , along with their colors, in the order dictated by  $\pi_k$ . Also,  $P_k$  adds the respective edges to previously exposed vertices when the current vertex is exposed to satisfy the basic requirement of VA model. This is possible because all players know the graph  $G$  and the random partition of the entries of matrix  $M$  among  $p$  players. After exposing all the vertices in  $V_k$ , as described,  $P_k$  sends the current memory state to player  $P_{k+1}$ . Assume that  $P_1 = P_{p+1}$ .

**Round  $r$ :** All the players behave similarly as in the previous rounds, except that, the player  $P_p$  does not send the current memory state to  $P_1$ . Rather,  $P_p$  decides whether there is a column in  $M$  with all 1s if the streaming algorithm  $\mathcal{S}$  decides that there are  $\Omega(T)$  many monochromatic edges in  $G$ . Otherwise, if  $\mathcal{S}$  decides that there is no monochromatic edge in  $G$ , then  $P_p$  decides that all the columns of  $M$  have weight either 0 or 1. Then  $P_p$  sends the output to all other players.

The vertices of graph  $G$  are indeed exposed randomly to the streaming algorithm. It is because the entries of matrix  $M$  are randomly partitioned among the players and each player also generates a random permutation of the vertices corresponding to the entries of matrix  $M$  available to them. From the description of the protocol  $\mathcal{A}$ , the memory state of the streaming algorithm (of space complexity  $s$ ) is communicated  $(r - 1)p + (p - 1)$  times and  $p - 1$  bits is communicated at the end by player  $P_p$  to broadcast the output. Hence, the communication cost of the protocol  $\mathcal{A}$  is at most  $O(rsp)$ .

Now we are left to prove the correctness of the protocol  $\mathcal{A}$ . If there is a column in  $M$  with all 1s, then all the vertices corresponding to entries of that column are colored with color  $C_*$ . Recall that there is a one-to-one correspondence between the columns in matrix  $M$  and cliques in the graph  $G$ . So, all the vertices of the clique, corresponding to the column having all 1s, are colored with the color  $C_*$ . As the size of each clique in the graph  $G$  is  $\sqrt{T}$ , there are at most  $\Omega(T)$  monochromatic edges. To prove the converse, assume that there is no column in the matrix  $M$  having all 1s. By UNIQUE INTERSECTION PROMISE, all the



## 15:12 Even the Easiest(?) Graph Coloring Problem Is Not Easy in Streaming!

columns have hamming weight at most 1. We will argue that there is no monochromatic edge in  $G$ . Consider an edge  $e$  in  $G$ . By the structure of  $G$ , the two vertices of  $e$  must be in the same clique, say the  $j$ -th clique, that is, let  $e = \{v_{i_1j}, v_{i_2j}\}$ . By the coloring scheme used by the protocols,  $v_{i_1j}$  and  $v_{i_2j}$  are colored according to the values of  $M_{i_1j}$  and  $M_{i_2j}$ , respectively. Note that both  $M_{i_1j}$  and  $M_{i_2j}$  belong to  $j$ -th column. As the hamming weight of every column is at most 1, there are three possibilities:

- (i)  $M_{i_1j} = M_{i_2j} = 0$ , that is,  $v_{i_1j}$  and  $v_{i_2j}$  are colored with color  $C_{i_1}$  and  $C_{i_2}$ , respectively;
- (ii)  $M_{i_1j} = 0$  and  $M_{i_2j} = 1$ , that is,  $v_{i_1j}$  and  $v_{i_2j}$  are colored with color  $C_{i_1}$  and  $C_*$ , respectively;
- (iii)  $M_{i_1j} = 1$  and  $M_{i_2j} = 0$ , that is,  $v_{i_1j}$  and  $v_{i_2j}$  are colored with color  $C_*$  and  $C_{i_2}$ , respectively.

In any case, the edge  $e = \{v_{i_1j}, v_{i_2j}\}$  is not monochromatic. This establishes the correctness of protocol  $\mathcal{A}$  for  $\text{DISJOINTNESS}_R\left(\sqrt{T}, \frac{n}{\sqrt{T}}, p\right)$ . ◀

### 5 CONFLICT-SEP in VARAND model

Using a structural property of the graph, we design a simple algorithm to solve the CONFLICT-SEP problem in the VARAND model.

► **Theorem 5.1.** *Given any graph  $G = (V, E)$  and a coloring function  $f : V(G) \rightarrow [C]$  and a parameter  $\varepsilon > 0$  as input, there exists an algorithm that solves the CONFLICT-SEP problem in the VARAND streaming model using space  $\tilde{O}\left(\frac{|V|}{\sqrt{\varepsilon|E|}}\right)$  with high probability.*

Let  $G'$  denote the subgraph of  $G$  consisting of only monochromatic edges in  $G$ . The lemma stated below guarantees that either there exists a large matching of size at least  $\sqrt{\varepsilon m}$  in  $G'$  or there exists a vertex of degree at least  $\sqrt{\varepsilon m}$  in  $G'$ .

► **Lemma 5.2** ([16]). *Let  $G = (V, E)$  be a graph and  $f : V(G) \rightarrow [C]$  be a coloring function such that at least  $\varepsilon$  fraction of the edges of  $E(G)$  are known to be monochromatic. Then, either there is a matching of size at least  $\sqrt{\varepsilon m}$  or there exists a vertex of degree at least  $\sqrt{\varepsilon m}$  in the subgraph  $G'$  defined on the monochromatic edges of  $G$ .*

The algorithm is as simple as it can get. We sample independently and uniformly at random the vertices in stream with probability  $p = \min\{1, \frac{10 \log n}{\sqrt{m}}\}$ <sup>7</sup> and store these vertices along with their colors. Let  $S \subseteq V$  be the set of sampled vertices. When a vertex appears in a stream, we check if it forms a monochromatic edge with one of the stored vertices in  $S$ . At the end of the stream, the algorithm declares the graph to be properly colored (valid) if it can not find a monochromatic edge, else it declares the instance to be  $\varepsilon$ -far from being monochromatic.

We show that Theorem 5.1 follows easily using Lemma 5.2.

<sup>7</sup> For simplicity of presentation, we assumed that, the number of edges  $m$  in graph  $G$  is known before the stream starts. However, this assumption can be removed by a simple tweak of starting with a value of  $m$  and increasing it in stages and adjusting the random sample accordingly. This is common in streaming algorithms.

**Proof.** We consider the following two cases.

- Case 1 – There exists a matching of size at least  $\sqrt{\varepsilon m}$ : Note that all these matched edges are monochromatic. Let  $(u, v)$  denote an arbitrary matched edge where  $u$  appears in the stream before  $v$ . Now, the edge  $(u, v)$  will be detected as monochromatic if vertex  $u$  has been sampled by the algorithm. The probability that vertex  $u$  is sampled is  $\frac{10 \log n}{\sqrt{m}}$ . Since, there are  $\sqrt{\varepsilon m}$  matched monochromatic edges, the algorithm will detect at least one of these matched monochromatic edges with probability at least  $(1 - 1/n^2)$ .
- Case 2 – There exists a vertex of degree at least  $\sqrt{\varepsilon m}$ : In this case most of the monochromatic edges may be incident on very few high degree vertices. To detect these edges, we want to store either the high degree vertices or one of its neighbours. But, if these high degree vertices appear at the beginning of the stream and we fail to sample them, then we may not detect a monochromatic edge. This is where the *random order* of vertices arriving in the stream comes into play. Now, assuming random order of vertices in the stream, at least  $\frac{1}{5}\sqrt{\varepsilon m}$  neighbors of  $v$  should appear before  $v$  in the stream with probability at least  $(1 - e^{-\frac{9}{50}\sqrt{\varepsilon m}})$ . Since we sample every vertex with probability  $\frac{10 \log n}{\sqrt{m}}$ , with high probability at least  $(1 - 1/n^2)$  one of its neighbors will be stored. ◀

## 6 Conclusion and Discussion

In this paper, we introduced a graph coloring problem to streaming setting with a different flavor – the coloring function streams along with the graph. We study the problem of CONFLICT-EST (estimating the number of monochromatic edges) and CONFLICT-SEP (detecting a separation between the number of valid edges) in VA, VADEG, and VARAND models. Our algorithms for VA and VADEG are tight upto polylogarithmic factors. However, a matching lower bound on the space complexity for VARAND model is still elusive. There is a gap between our upper and lower bound results for VARAND model in terms of the exponent in  $T$ . Our hunch is that the upper bound is tight. Specifically, we obtained an upper bound of  $\tilde{O}\left(\frac{n}{\sqrt{T}}\right)$  and the lower bound is  $\Omega\left(\frac{n}{T^2}\right)$ . Here we would like to note that the lower bound also holds in AL and VADEG model when the vertices are exposed in a random order. However, we feel that our algorithm for CONFLICT-EST in VARAND model is tight upto polylogarithmic factors. We leave this problem open.

We feel the *edge coloring* counterpart of the vertex coloring problem proposed in the paper will be worthwhile to study. Let the edges of  $G$  be colored with a function  $f : E(G) \rightarrow [C]$ , for  $C \in \mathbb{N}$ . A vertex  $u \in V(G)$  is said to be a *validly* colored vertex if no two edges incident on  $u$  have the same color. An edge coloring is valid if all vertices are validly colored. Consider the AL model for the edge coloring problem. As all edges incident on an exposed vertex  $u$  are revealed in the stream, if we can solve a duplicate element finding problem on the colors of the edges incident on  $u$ , then we are done! It seems at a first glance that all the three models of VA, AL and EA will be difficult to handle for the edge coloring problem on streams of graph and edge colors. It would be interesting to see if the edge coloring variant of the problems we considered in this paper, admit efficient streaming algorithms. We plan to look at this problem next.

---

## References

- 1 Noga Alon and Sepehr Assadi. Palette sparsification beyond  $(\Delta+1)$  vertex coloring. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2020, August 17-19, 2020, Virtual Conference*, volume 176 of *LIPIcs*, pages 6:1–6:22. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.APPROX/RANDOM.2020.6.

## 15:14 Even the Easiest(?) Graph Coloring Problem Is Not Easy in Streaming!

- 2 Noga Alon and Sepehr Assadi. Palette sparsification beyond  $(\Delta+1)$  vertex coloring. *CoRR*, abs/2006.10456, 2020. [arXiv:2006.10456](#).
- 3 Sepehr Assadi, Yu Chen, and Sanjeev Khanna. Sublinear algorithms for  $(\Delta + 1)$  vertex coloring. In Timothy M. Chan, editor, *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 767–786. SIAM, 2019. doi:10.1137/1.9781611975482.48.
- 4 Soheil Behnezhad, Mahsa Derakhshan, MohammadTaghi Hajiaghayi, Marina Knittel, and Hamed Saleh. Streaming and massively parallel algorithms for edge coloring. In Michael A. Bender, Ola Svensson, and Grzegorz Herman, editors, *27th Annual European Symposium on Algorithms, ESA 2019, September 9-11, 2019, Munich/Garching, Germany*, volume 144 of *LIPIcs*, pages 15:1–15:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.ESA.2019.15.
- 5 Suman K. Bera and Amit Chakrabarti. Towards tighter space bounds for counting triangles and other substructures in graph streams. In Heribert Vollmer and Brigitte Vallée, editors, *34th Symposium on Theoretical Aspects of Computer Science, STACS 2017, March 8-11, 2017, Hannover, Germany*, volume 66 of *LIPIcs*, pages 11:1–11:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPIcs.STACS.2017.11.
- 6 Suman K. Bera, Amit Chakrabarti, and Prantar Ghosh. Graph coloring via degeneracy in streaming and other space-conscious models. *CoRR*, abs/1905.00566, 2019. [arXiv:1905.00566](#).
- 7 Suman K. Bera and C. Seshadhri. How the degeneracy helps for triangle counting in graph streams. In Dan Suciu, Yufei Tao, and Zhewei Wei, editors, *Proceedings of the 39th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2020, Portland, OR, USA, June 14-19, 2020*, pages 457–467. ACM, 2020. doi:10.1145/3375395.3387665.
- 8 Suman Kalyan Bera and Prantar Ghosh. Coloring in graph streams. *CoRR*, abs/1807.07640, 2018. [arXiv:1807.07640](#).
- 9 Anup Bhattacharya, Arijit Bishnu, Gopinath Mishra, and Anannya Upasana. Even the easiest(?) graph coloring problem is not easy in streaming! *CoRR*, abs/2010.13143, 2020. [arXiv:2010.13143](#).
- 10 Amit Chakrabarti, Graham Cormode, and Andrew McGregor. Robust lower bounds for communication and stream computation. *Theory Comput.*, 12(1):1–35, 2016. doi:10.4086/toc.2016.v012a010.
- 11 Graham Cormode, Jacques Dark, and Christian Konrad. Independent sets in vertex-arrival streams. In Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi, editors, *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*, volume 132 of *LIPIcs*, pages 45:1–45:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.ICALP.2019.45.
- 12 Devdatt P. Dubhashi and Alessandro Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms*. Cambridge University Press, 2009. URL: <http://www.cambridge.org/gb/knowledge/isbn/item2327542/>.
- 13 Guy Even, Magnús M. Halldórsson, Lotem Kaplan, and Dana Ron. Scheduling with conflicts: online and offline algorithms. *J. Sched.*, 12(2):199–224, 2009. doi:10.1007/s10951-008-0089-1.
- 14 Uriel Feige and Joe Kilian. Zero knowledge and the chromatic number. *J. Comput. Syst. Sci.*, 57(2):187–199, 1998. doi:10.1006/jcss.1998.1587.
- 15 M. R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.
- 16 Stasys Jukna. *Extremal Combinatorics - With Applications in Computer Science*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2011. doi:10.1007/978-3-642-17364-6.

- 17 John Kallaughner, Michael Kapralov, and Eric Price. The sketching complexity of graph and hypergraph counting. In Mikkel Thorup, editor, *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 556–567. IEEE Computer Society, 2018. doi:10.1109/FOCS.2018.00059.
- 18 John Kallaughner, Andrew McGregor, Eric Price, and Sofya Vorotnikova. The complexity of counting cycles in the adjacency list streaming model. In Dan Suciu, Sebastian Skritek, and Christoph Koch, editors, *Proceedings of the 38th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2019, Amsterdam, The Netherlands, June 30 - July 5, 2019*, pages 119–133. ACM, 2019. doi:10.1145/3294052.3319706.
- 19 Daniel M. Kane, Kurt Mehlhorn, Thomas Sauerwald, and He Sun. Counting arbitrary subgraphs in data streams. In Artur Czumaj, Kurt Mehlhorn, Andrew M. Pitts, and Roger Wattenhofer, editors, *Automata, Languages, and Programming - 39th International Colloquium, ICALP 2012, Warwick, UK, July 9-13, 2012, Proceedings, Part II*, volume 7392 of *Lecture Notes in Computer Science*, pages 598–609. Springer, 2012. doi:10.1007/978-3-642-31585-5\_53.
- 20 Subhash Khot and Ashok Kumar Ponnuswami. Better inapproximability results for maxclique, chromatic number and min-3lin-deletion. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *Automata, Languages and Programming, 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10-14, 2006, Proceedings, Part I*, volume 4051 of *Lecture Notes in Computer Science*, pages 226–237. Springer, 2006. doi:10.1007/11786986\_21.
- 21 Eyal Kushilevitz and Noam Nisan. *Communication complexity*. Cambridge University Press, 1997.
- 22 Andrew McGregor. Graph stream algorithms: a survey. *SIGMOD Rec.*, 43(1):9–20, 2014. doi:10.1145/2627692.2627694.
- 23 Andrew McGregor, Sofya Vorotnikova, and Hoa T. Vu. Better algorithms for counting triangles in data streams. In Tova Milo and Wang-Chiew Tan, editors, *Proceedings of the 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2016, San Francisco, CA, USA, June 26 - July 01, 2016*, pages 401–411. ACM, 2016. doi:10.1145/2902251.2902283.
- 24 Wolfgang Mulzer. Five proofs of chernoff’s bound with applications. *Bull. EATCS*, 124, 2018. URL: <http://eatcs.org/beatcs/index.php/beatcs/article/view/525>.
- 25 Isabelle Stanton and Gabriel Kliot. Streaming graph partitioning for large distributed graphs. In Qiang Yang, Deepak Agarwal, and Jian Pei, editors, *The 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ’12, Beijing, China, August 12-16, 2012*, pages 1222–1230. ACM, 2012. doi:10.1145/2339530.2339722.
- 26 Charalampos E. Tsourakakis, Christos Gkantsidis, Bozidar Radunovic, and Milan Vojnovic. FENNEL: streaming graph partitioning for massive scale graphs. In Ben Carterette, Fernando Diaz, Carlos Castillo, and Donald Metzler, editors, *Seventh ACM International Conference on Web Search and Data Mining, WSDM 2014, New York, NY, USA, February 24-28, 2014*, pages 333–342. ACM, 2014. doi:10.1145/2556195.2556213.
- 27 V. G. Vizing. On an estimate of the chromatic class of a p-graph, 1964.
- 28 David Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. *Theory of Computing*, 3(1):103–128, 2007. doi:10.4086/toc.2007.v003a006.

**A** Some probability results

► **Lemma A.1** ([12], Chernoff-Hoeffding bound). *Let  $X_1, \dots, X_n$  be independent random variables such that  $X_i \in [0, 1]$ . For  $X = \sum_{i=1}^n X_i$  and  $\mu = \mathbb{E}[X]$ , the following holds for any  $0 \leq \delta \leq 1$ :*

$$\mathbb{P}(|X - \mu| \geq \delta\mu) \leq 2 \exp\left(-\frac{\mu\delta^2}{3}\right)$$

► **Lemma A.2** ([24]). *Let  $I = \{1, \dots, N\}$ ,  $r \in [N]$  be a given parameter. If we sample a subset  $R$  without replacement, then the following holds for any  $J \subset I$  and  $\delta \in (0, 1)$ .*

- (i)  $\mathbb{P}(|J \cap R| \geq (1 + \delta)|J| \frac{r}{N}) \leq \exp\left(-\frac{\delta^2 |J| r}{3N}\right);$
- (ii)  $\mathbb{P}(|J \cap R| \leq (1 - \delta)|J| \frac{r}{N}) \leq \exp\left(-\frac{\delta^2 |J| r}{3N}\right);$
- (iii) *Further more, we have the following if  $|J| \leq k$ , then the following holds.*

$$\mathbb{P}(|J \cap R| \geq (1 + \delta)k \frac{r}{N}) \leq \exp\left(-\frac{\delta^2 k r}{3N}\right)$$

**B** Proof of Lemma 3.2

► **Lemma B.1** (Restatement of Lemma 3.2).

- (i) *For each  $j \in [t]$  with  $|B_j| \geq \frac{\sqrt{\varepsilon T}}{10t}$ ,  $\mathbb{P}\left(\left||B_j \cap R| - \frac{|R||B_j|}{n}\right| \geq \frac{\varepsilon}{10} \frac{|R||B_j|}{n}\right) \leq \frac{1}{n^{10}}.$*
- (ii) *For each  $j \in [t]$  with  $|B_j| < \frac{\sqrt{\varepsilon T}}{10t}$ ,  $\mathbb{P}\left(|B_j \cap R| \geq \frac{|R|}{n} \frac{\sqrt{\varepsilon T}}{8t}\right) \leq \frac{1}{n^{10}}.$*
- (iii) *For each vertex  $v_i$  with  $d_M(v_i) \geq \frac{\sqrt{\varepsilon T}}{10t}$ ,  $\mathbb{P}\left(\left|\kappa_{v_i} - \frac{|R|d_M(v_i)}{n}\right| \geq \frac{\varepsilon}{10} \frac{|R|d_M(v_i)}{n}\right) \leq \frac{1}{n^{10}}.$*
- (iv) *For each vertex  $v_i$  with  $d_M(v_i) < \frac{\sqrt{\varepsilon T}}{10t}$ ,  $\mathbb{P}\left(\kappa_{v_i} \geq \frac{|R|}{n} \frac{\sqrt{\varepsilon T}}{8t}\right) \leq \frac{1}{n^{10}}.$*

**Proof.** Let us take  $N = n, r = |R| = \Gamma = \tilde{\Theta}\left(\frac{n}{\sqrt{T}}\right)$ ,  $I = \{v_1, \dots, v_n\}$  in Lemma A.2.

- (i) Setting  $J = B_j$  and  $\delta = \frac{\varepsilon}{10}$  in Lemma A.2 (i) and (ii), we have

$$\mathbb{P}\left(\left||B_j \cap R| - \frac{|R||B_j|}{n}\right| \geq \frac{\varepsilon}{10} \frac{|R||B_j|}{n}\right) \leq 2 \exp\left(-\frac{(\varepsilon/10)^2 |B_j| \Gamma}{3n}\right) \leq \frac{1}{n^{10}}.$$

The last inequality holds as  $|B_j| \geq \frac{\sqrt{\varepsilon T}}{10t}$ ,  $t = \lceil \log_{1+\frac{\varepsilon}{10}} n \rceil = \Theta\left(\frac{\log n}{\varepsilon}\right)$  and  $\Gamma = \tilde{\Theta}\left(\frac{n}{\sqrt{T}}\right)$ .

- (ii) Set  $J = B_j$ ,  $k = \frac{\sqrt{\varepsilon T}}{10t}$ ,  $\delta = \frac{1}{4}$  in Lemma A.2 (iii). As  $|B_j| \leq \frac{\sqrt{\varepsilon T}}{10t}$ ,  $|J| \leq k$ . Hence,

$$\mathbb{P}\left(|B_j \cap R| \geq \frac{|R|}{n} \frac{\sqrt{\varepsilon T}}{8t}\right) \leq \exp\left(-\frac{(1/4)^2 (\sqrt{\varepsilon T}/10t) \Gamma}{3n}\right) \leq \frac{1}{n^{10}}.$$

- (iii) Setting  $J$  as the set of monochromatic neighbors of  $v_i$  in  $R$  and  $\delta = \frac{\varepsilon}{10}$  in Lemma A.2 (i) and (ii), we get

$$\mathbb{P}\left(\left|\kappa_{v_i} - \frac{|R|d_M(v_i)}{n}\right| \geq \frac{\varepsilon}{10} \frac{|R|d_M(v_i)}{n}\right) \leq \exp\left(-\frac{(\varepsilon/10)^2 |J| \Gamma}{3n}\right) \leq \frac{1}{n^{10}}.$$

The last inequality holds as  $|J| = d_M(v_i) \geq \frac{\sqrt{\varepsilon T}}{10t}$ ,  $t = \lceil \log_{1+\frac{\varepsilon}{10}} n \rceil = \Theta\left(\frac{\log n}{\varepsilon}\right)$  and  $\Gamma = \tilde{\Theta}\left(\frac{n}{\sqrt{T}}\right)$ .

- (iv) Set  $J$  as the set of monochromatic neighbors of  $v_i$  in  $R$ ,  $k = \frac{\sqrt{\varepsilon T}}{10t}$ ,  $\delta = \frac{1}{4}$  in Lemma A.2 (iii). Note that  $|J| = d_M(v_i) \leq \frac{\sqrt{\varepsilon T}}{10t} = k$ . Hence,

$$\mathbb{P}\left(\kappa_{v_i} \geq \frac{|R|}{n} \frac{\sqrt{\varepsilon T}}{8t}\right) \leq \exp\left(-\frac{(1/4)^2(\sqrt{\varepsilon T}/10t)\Gamma}{3n}\right) \leq \frac{1}{n^{10}}. \quad \blacktriangleleft$$

## C Communication Complexity

Communication Complexity [21] deals with finding the minimum amount of space that is needed to communicate in order to compute a function when the input to the function is distributed among multiple parties. For the purpose of our work, we are concerned with two player games with one-way communication protocol. The players are traditionally called Alice and Bob. Both of them have a  $n$ -bit input string and are unaware of each other's input. The goal is to minimize the bits Alice needs to communicate to Bob so that he can compute a function on both their inputs. No assumption is made on their computational powers and there is no restriction on the amount of time needed for computing the function.

### C.1 INDEX problem in the communication complexity model

Lower bound results in the streaming model of computation are proved by reduction from a problem in communication complexity model. We determine our lower bounds by considering a reduction from the INDEX problem in the one-way communication protocol for two players to the specific problem in graphs in the VA model. The INDEX problem is defined as follows: There are two parties, Alice and Bob. Alice has a  $N$ -bit input string  $X \in \{0, 1\}^N$  and Bob has an integer  $j \in [N]$ . Both are unaware of each other's input and the goal is to compute  $X_j$ , the  $j^{\text{th}}$  bit of  $X$ . The lower bound for space complexity to solve the INDEX problem in the one-way deterministic communication model is  $\Omega(N)$ .

► **Lemma C.1** ([21]). *The deterministic communication complexity of INDEX is  $\Omega(N)$*

## D Algorithm for CONFLICT-EST in VARAND model

- $\Gamma = \tilde{\Theta}\left(\frac{n}{\sqrt{T}}\right)$ ;  $v_1, \dots, v_n$  be the random ordering in which vertices are revealed and  $R = \{v_1, \dots, v_\Gamma\}$ ;
- $\kappa_{v_i}, i \in [n]$ , denotes the number of monochromatic neighbors of  $v_i$  in  $R$ ,
- $\widehat{d}_{v_i}, i \in [n]$ , denotes the (estimated) monochromatic neighbors of vertices in  $G$ .
- $H$  denotes the set of *high* degree vertex in  $R$ , i.e.,  $H = \{v_i : \kappa_{v_i} \geq \frac{|R|}{n} \frac{\sqrt{\varepsilon T}}{8t}\}$  and  $L = V \setminus H$ ;  $L_R = L \cap R$  and  $H_R = H \cap R$ ;
- The vertices in  $L$  are partitioned into  $t$  buckets as follows:  
 $B_j = \{v_i \in L : (1 + \frac{\varepsilon}{10})^{j-1} \leq d_M(v_i) < (1 + \frac{\varepsilon}{10})^j\}$ , where  $j \in [t]$ .

■ **Algorithm 1** RANDOM-ORDER-EST( $\varepsilon, T$ ): CONFLICT-EST in VARAND model.

---

**Input:**  $G = (V, E)$  and a coloring function  $f$  on  $V$  in the VARAND model, parameters  $T$  and  $\varepsilon$ .

**Output:**  $\widehat{m}$ , that is, a  $(1 \pm \varepsilon)$  approximation to  $|E_M|$ .

Set  $t = \lceil \log_{1+\frac{\varepsilon}{10}} n \rceil$ . If  $T < 63t^2$ , then store all the vertices in  $G$  along with their colors. At the end, report the exact value of  $|E_M|$ . Otherwise, we proceed through via three building blocks described below and marked as (1),(2), (3) and (4). Refer to the notations described above this pseudocode.

(1) Processing the vertices in  $R$ , the first  $\Gamma$  vertices, in the stream:

**for** (each vertex  $v_i \in R$  exposed in the stream) **do**  
 | Store  $v_i$  as well as its color  $f(v_i)$ .  
 | For each edge  $(v_{i'}, v_i)$  that arrives in the stream, increase the values of  $\kappa_{v_{i'}}$  and  $\kappa_{v_i}$ .  
**end**

(2) Computation of some parameters based on vertices in  $R$  and their colors:

**for** (each  $v_i \in R$  with  $\kappa_{v_i} \geq \frac{|R|}{n} \frac{\sqrt{\varepsilon T}}{8t}$ ) **do**  
 | Add  $v_i$  to  $H_R$ , and set  $\widehat{d}_{v_i} = \frac{n}{|R|} \kappa_{v_i}$ .  
**end**  
 $\widehat{m}_h = \sum_{v_i \in H} \widehat{d}_{v_i}$ .  
 Let  $L_R = R \setminus H_R$ .  
**for** (each  $v_i \in L_R$ ) **do**  
 | Set  $\widehat{d}_{v_i} = \kappa_{v_i}$ .  
**end**

(3) Processing the vertices in  $V(G) \setminus R$  in the stream:

**for** (each vertex  $v_i \notin R$  exposed in the stream) **do**  
 | Determine the value of  $\kappa_{v_i}$ . If  $\kappa_{v_i} \geq \frac{|R|}{n} \frac{\sqrt{\varepsilon T}}{8t}$ , find  $\widehat{d}_{v_i} = \frac{n}{|R|} \kappa_{v_i}$  and add  $\widehat{d}_{v_i}$  to the current  $\widehat{m}_h$ .  
 | Also, for each  $v_{i'} \in L_R$ , increase the value of  $\widehat{d}_{v_{i'}}$  if  $(v_{i'}, v_i)$  is an edge.  
**end**

(4) Post processing, after the stream ends, to return the output:

From the values of  $\widehat{d}_{v_i}$  for all  $v_i \in L_R$ , determine the buckets for each vertex in  $L_R$ . Also, for each  $j \in [t]$ , find  $|A_j| = |L_R \cap B_j|$ . Then determine

$$\widehat{m}_\ell = \frac{n}{|R|} \sum_{j \in [t]} |A_j| \left(1 + \frac{\varepsilon}{10}\right)^j.$$

Report  $\widehat{m} = \frac{\widehat{m}_h + \widehat{m}_\ell}{2}$  as the final OUTPUT.

---



**E** Algorithm for CONFLICT-SEP in VARAND model**Algorithm 2** Algorithm: CONFLICT-SEP in VERTEX ARRIVAL IN RANDOM ORDER model**Input:**  $G = (V, E)$  and a coloring function  $f$  on  $V$  in the VARAND model**Output:** The algorithm verifies if  $f$  is  $\varepsilon$ -far from valid or notLet  $S$  be the set of stored vertices and their colors. Initially,  $S$  is empty. **for**  $i \leftarrow 1$  **to**  $|V|$  **do**    let  $u$  be the  $i^{th}$  vertex of the stream    Store  $u$  and its color  $f(u)$  in  $S$  with probability  $\mathcal{O}\left(\frac{\log n}{\sqrt{m}}\right)$     **for every vertex**  $v$  **in**  $S$  **do**        Check if  $(v, u)$  is an edge and  $f(v) = f(u)$     **end****end**Output  $f$  is valid if none of the edges sampled are conflicting, else output that  $f$  is  $\varepsilon$ -far from being valid.



# The Variable-Processor Cup Game

**William Kuszmaul**

CSAIL, MIT, Cambridge, MA, USA  
kuszmaul@mit.edu

**Alek Westover**

CSAIL, MIT, Cambridge, MA, USA  
alek.westover@gmail.com

---

## Abstract

The problem of scheduling tasks on  $p$  processors so that no task ever gets too far behind is often described as a game with cups and water. In the  $p$ -processor cup game on  $n$  cups, there are two players, a filler and an emptier, that take turns adding and removing water from a set of  $n$  cups. In each turn, the filler adds  $p$  units of water to the cups, placing at most 1 unit of water in each cup, and then the emptier selects  $p$  cups to remove up to 1 unit of water from. The emptier's goal is to minimize the backlog, which is the height of the fullest cup.

The  $p$ -processor cup game has been studied in many different settings, dating back to the late 1960's. All of the past work shares one common assumption: that  $p$  is fixed. This paper initiates the study of what happens when the number of available processors  $p$  varies over time, resulting in what we call the *variable-processor cup game*.

Remarkably, the optimal bounds for the variable-processor cup game differ dramatically from its classical counterpart. Whereas the  $p$ -processor cup has optimal backlog  $\Theta(\log n)$ , the variable-processor game has optimal backlog  $\Theta(n)$ . Moreover, there is an efficient filling strategy that yields backlog  $\Omega(n^{1-\epsilon})$  in quasi-polynomial time against any deterministic emptying strategy.

We additionally show that straightforward uses of randomization cannot be used to help the emptier. In particular, for any positive constant  $\Delta$ , and any  $\Delta$ -greedy-like randomized emptying algorithm  $\mathcal{A}$ , there is a filling strategy that achieves backlog  $\Omega(n^{1-\epsilon})$  against  $\mathcal{A}$  in quasi-polynomial time.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Parallel algorithms

**Keywords and phrases** scheduling, cup games, online algorithms, lower bounds

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.16

**Related Version** <https://arxiv.org/abs/2012.00127>

**Funding** This research was sponsored in part by National Science Foundation Grants XPS-1533644 and CCF-1930579, and in part by the United States Air Force Research Laboratory and was accomplished under Cooperative Agreement Number FA8750-19-2-1000. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the United States Air Force or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

*William Kuszmaul:* Funded by a Hertz Fellowship and an NSF GRFP fellowship.

**Acknowledgements** Portions of this work were completed at the Second Hawaii Workshop on Parallel Algorithms and Data Structures. The authors would like to thank Nodari Sitchinava for organizing the workshop, and John Iacono for several helpful discussions relating to the variable-processor cup game.

## 1 Introduction

A fundamental challenge in processor scheduling is how to perform load balancing, that is, how to share processors among tasks in order to keep any one task from getting too far behind. Consider  $n$  tasks executing in time slices on  $p < n$  processors. During each time slice,



© William Kuszmaul and Alek Westover;  
licensed under Creative Commons License CC-BY  
12th Innovations in Theoretical Computer Science Conference (ITCS 2021).  
Editor: James R. Lee; Article No. 16; pp. 16:1–16:20



Leibniz International Proceedings in Informatics  
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

a scheduler must select  $p$  (distinct) tasks that will be executed during the slice; up to one unit of work is completed on each executed task. During the same time slice, however, up to  $p$  units of *new work* may be allocated to the tasks, with different tasks receiving different amounts of work. The goal of a load-balancing scheduler is to bound the backlog of the system, which is defined to be the maximum amount of uncompleted work for any task.

As a convention, the load balancing problem is often described as a game involving water and cups. The  **$p$ -processor cup game** is a multi-round game with two players, an **emptier** and a **filler**, that takes place on  $n$  initially empty cups. At the beginning of each round, the filler adds up to  $p$  units of water to the cups, subject to the constraint that each cup receives at most 1 unit of water. The emptier then selects up to  $p$  distinct cups and removes up to 1 unit of water from each of them. The emptier's goal is to minimize the amount of water in the fullest cup, also known as the **backlog**. In terms of processor scheduling, the cups represent tasks, the water represents work assigned to each task, and the emptier represents a scheduling algorithm.

Starting with the seminal paper of Liu [31], work on the  $p$  processor cup game has spanned more than five decades [7, 21, 8, 30, 28, 34, 6, 24, 31, 32, 17, 10, 26, 1, 16, 29]. In addition to processor scheduling [7, 21, 8, 30, 28, 34, 6, 24, 31, 32, 1, 29, 17], applications include network-switch buffer management [22, 4, 36, 20], quality of service guarantees [7, 1, 29], and data structure deamortization [2, 17, 16, 3, 35, 23, 18, 25, 9].

The game has also been studied in many different forms. Researchers have studied the game with a fixed-filling-rate constraint [7, 21, 8, 30, 28, 34, 6, 24, 31, 32], with various forms of resource augmentation [10, 26, 29, 17], with both oblivious and adaptive adversaries [1, 7, 31, 10, 26, 11], with smoothed analysis [26, 10], with a semi-clairvoyant emptier [29], with competitive analysis [5, 19, 15], etc.

For the plain form of the  $p$ -processor cup game, the greedy emptying algorithm (i.e., always empty from the fullest cups) is known to be asymptotically optimal [1, 10, 26], achieving backlog  $O(\log n)$ . The optimal backlog for randomized emptying algorithms remains an open question [17, 10, 26] and is known to be between  $\Omega(\log \log n)$  and  $O(\log \log n + \log p)$  [26].

### This paper: varying resources

Although cup games have been studied in many forms, all of the prior work on cup games shares one common assumption: the number  $p$  of processors is fixed.

In modern computing, however, computers are often shared among multiple applications, users, and even virtual OS's. The result is that the amount of resources (e.g., memory, processors, cache, etc.) available to a given application may fluctuate over time. The problem of handling cache fluctuations has received extensive attention in recent years (see work on cache-adaptive analysis [33, 12, 13, 14, 27]), but the problem of handling a varying number of processors remains largely unstudied.

This paper introduces the **variable-processor cup game**, in which the filler is allowed to *change*  $p$  (the total amount of water that the filler adds, and the emptier removes, from the cups per round) at the beginning of each round. Note that we do not allow the resources of the filler and emptier to vary separately. That is, as in the standard game, we take the value of  $p$  for the filler and emptier to be identical in each round. This restriction is crucial since, if the filler is allowed more resources than the emptier, then the filler could trivially achieve unbounded backlog.

A priori having variable resources offers neither player a clear advantage. When the number  $p$  of processors is fixed, the greedy emptying algorithm (i.e., always empty from the fullest cups), is known to achieve backlog  $O(\log n)$  [1, 10, 26] regardless of the value of  $p$ .

This seems to suggest that, when  $p$  varies, the correct backlog should remain  $O(\log n)$ . In fact, when we began this project, we hoped for a straightforward reduction between the two versions of the game.

## Results

We show that the variable-processor cup game is *not* equivalent to the standard  $p$ -processor game. By strategically controlling the number  $p$  of processors, the filler can achieve substantially larger backlog than would otherwise be possible.

We begin by constructing filling strategies against deterministic emptying algorithms. We show that for any positive constant  $\varepsilon$ , there is a filling strategy that achieves backlog  $\Omega(n^{1-\varepsilon})$  within  $2^{\text{polylog}(n)}$  rounds. Moreover, if we allow for  $n!$  rounds, then there is a filling strategy that achieves backlog  $\Omega(n)$ . In contrast, for the  $p$ -processor cup game with any fixed  $p$ , the backlog never exceeds  $O(\log n)$ .

Our lower-bound construction is asymptotically optimal. By introducing a novel set of invariants, we deduce that the greedy emptying algorithm never lets backlog exceed  $O(n)$ .

A natural question is whether *randomized* emptying algorithms can do better. In particular, when the emptier is randomized, the filler is taken to be **oblivious**, meaning that the filler cannot see what the emptier does at each step. Thus the emptier can potentially use randomization to obfuscate their behavior from the filler, preventing the filler from being able to predict the heights of cups.

When studying randomized emptying strategies, we restrict ourselves to the class of **greedy-like emptying strategies**. This means that the emptier never chooses to empty from a cup  $c$  over another cup  $c'$  whose fill is more than  $\omega(1)$  greater than the fill of  $c$ . All of the known randomized emptying strategies for the classic  $p$ -processor cup game are greedy-like [10, 26].

Remarkably, the power of randomization does not help the emptier very much in the variable-processor cup game. For any constant  $\varepsilon > 0$ , we give an oblivious filling strategy that achieves backlog  $\Omega(n^{1-\varepsilon})$  in quasi-polynomial time (with probability  $1 - 2^{-\text{polylog } n}$ ), no matter what (possibly randomized) greedy-like strategy the emptier follows.

Our results combine to tell a surprising story. They suggest that the problem of varying resources poses a serious theoretical challenge for the design and analysis of load-balancing scheduling algorithms. There are many possible avenues for future work. Can techniques from beyond worst-case analysis (e.g., smoothing, resource augmentation, etc.) be used to achieve better bounds on backlog? What about placing restrictions on the filler (e.g., bounds on how fast  $p$  can change), allowing the emptier to be able to be semi-clairvoyant, or making stochastic assumptions on the filler? We believe that all of these questions warrant further attention.

## Paper outline

In Section 2 we establish the conventions and notations that we will use to discuss the variable-processor cup game. In Section 3 we analyze the greedy emptying algorithm, showing that it achieves backlog  $O(n)$ . We then turn our attention to designing (both oblivious and adaptive) filling strategies that achieve large backlog. Section 4 gives a technical overview of the filling strategies and their analyses. Section 5 then gives a full treatment of the filling strategies in the case where the filler is adaptive; the full treatment of the case where the filler is oblivious is deferred to the extended version of this paper.

## 2 Preliminaries

The cup game consists of a sequence of rounds. Let  $S_t$  denote the state of the cups at the start of round  $t$ . At the beginning of the round, the filler chooses the number of processors  $p_t$  for the round. Next, the filler distributes  $p_t$  units of water among the cups (with at most 1 unit of water to any particular cup). Now the game is at the **intermediate state** for round  $t$ , which we call state  $I_t$ . Finally the emptier chooses  $p_t$  cups to empty at most 1 unit of water from, which marks the conclusion of round  $t$ . The state is then  $S_{t+1}$ .

If the emptier empties from a cup  $c$  on round  $t$  such that the fill of  $c$  is less than 1 in state  $I_t$ , then  $c$  now has 0 fill (not negative fill); we say that the emptier **zeroes out**  $c$  on round  $t$ . Note that on any round where the emptier zeroes out a cup the emptier has removed less total fill from the cups than the filler has added to the cups; hence the average fill of the cups has increased.

We denote the fill of a cup  $c$  at state  $S$  by  $\text{fill}_S(c)$ . Let the **mass** of a set of cups  $X$  at state  $S$  be  $m_S(X) = \sum_{c \in X} \text{fill}_S(c)$ . Denote the average fill of a set of cups  $X$  at state  $S$  by  $\mu_S(X)$ .<sup>1</sup> Note that  $\mu_S(X)|X| = m_S(X)$ . Let the **backlog** at state  $S$  be  $\max_c \text{fill}_S(c)$ , let the **anti-backlog** at state  $S$  be  $\min_c \text{fill}_S(c)$ . Let the **rank** of a cup at a given state be its position in the list of the cups sorted by fill at the given state, breaking ties arbitrarily but consistently. For example, the fullest cup at a state has rank 1, and the least full cup has rank  $n$ . Let  $[n] = \{1, 2, \dots, n\}$ , let  $i + [n] = \{i + 1, i + 2, \dots, i + n\}$ . For a state  $S$ , let  $S(r)$  denote the rank  $r$  cup at state  $S$ , and let  $S(\{r_1, r_2, \dots, r_m\})$  denote the set of cups of ranks  $r_1, r_2, \dots, r_m$ .

As a tool in the analysis we define a new variant of the cup game: the **negative-fill** cup game. In the negative-fill cup game, when the emptier empties from a cup, the cup's fill always decreases by exactly 1, i.e. there is no zeroing out. We refer to the standard version of the cup game where cups can zero out as the **standard-fill** cup game.

The notion of negative fill will be useful in our lower-bound constructions, in which we want to construct a strategy for the filler that achieves large backlog. By analyzing a filling strategy on the negative-fill game, we can then reason about what happens if we apply the same filling strategy recursively to a set of cups  $S$  whose average fill  $\mu$  is larger than 0; in the recursive application, the average fill  $\mu$  acts as the “new 0”, and fills less than  $\mu$  act as negative fills.

Note that it is strictly easier for the filler to achieve high backlog in the standard-fill cup game than in the negative-fill cup game; hence a lower bound on backlog in the negative-fill cup game also serves as a lower bound on backlog in the standard-fill cup game. On the other hand, during the upper bound proof we use the standard-fill cup game: this makes it harder for the emptier to guarantee its upper bound.

### Other Conventions

When discussing the state of the cups **at a round  $t$** , we will take it as given that we are referring to the starting state  $S_t$  of the round. Also, when discussing sets, we will use  $XY$  as a shorthand for  $X \cup Y$ . Finally, when discussing the average fill  $\mu_{S_t}(X)$  of a set of cups, we will sometimes omit the subscript  $S_t$  when the round number is clear.

<sup>1</sup> For both mass and average fill, in cases where  $S$  is clear from context, we may omit the subscript, writing  $m(X)$  or  $\mu(X)$ .

### 3 Upper Bound

In this section we analyze the **greedy emptier**, which always empties from the  $p$  fullest cups. We prove in Corollary 2 that the greedy emptier prevents backlog from exceeding  $O(n)$ . In order to analyze the greedy emptier, we establish a system of invariants that hold at every step of the game.

The main result of the section is the following theorem.

► **Theorem 1.** *In the variable-processor cup game on  $n$  cups, the greedy emptier maintains, at every step  $t$ , the invariants*

$$\mu_{S_t}(S_t([k])) \leq 2n - k \quad (1)$$

for all  $k \in [n]$ .

By applying Theorem 1 to the case of  $k = 1$ , we arrive at a bound on backlog:

► **Corollary 2.** *In the variable-processor cup game on  $n$  cups, the greedy emptying strategy never lets backlog exceed  $O(n)$ .*

**Proof of Theorem 1.** We prove the invariants by induction on  $t$ . The invariants hold trivially for  $t = 1$  (the base case for the inductive proof): the cups start empty so  $\mu_{S_1}(S_1([k])) = 0 \leq 2n - k$  for all  $k \in [n]$ .

Fix a round  $t \geq 1$ , and any  $k \in [n]$ . We assume the invariant for all values of  $k' \in [n]$  for state  $S_t$  (we will only explicitly use two of the invariants for each  $k$ , but the invariants that we need depend on the choice of  $p_t$  by the filler) and show that the invariant on the  $k$  fullest cups holds on round  $t + 1$ , i.e. that

$$\mu_{S_{t+1}}(S_{t+1}([k])) \leq 2n - k.$$

Note that because the emptier is greedy it always empties from the cups  $I_t([p_t])$ . Let  $A$ , with  $a = |A|$ , be  $A = I_t([\min(k, p_t)]) \cap S_{t+1}([k])$ ;  $A$  consists of the cups that are among the  $k$  fullest cups in  $I_t$ , are emptied from, and are among the  $k$  fullest cups in  $S_{t+1}$ . Let  $B$ , with  $b = |B|$ , be  $B = I_t([\min(k, p_t)]) \setminus A$ ;  $B$  consists of the cups that are among the  $k$  fullest cups in state  $I_t$ , are emptied from, and are not among the  $k$  fullest cups in  $S_{t+1}$ . Let  $C = I_t(a + b + [k - a])$ , with  $c = k - a = |C|$ ;  $C$  consists of the cups with ranks  $a + b + 1, \dots, k + b$  in state  $I_t$ . The set  $C$  is defined so that  $S_{t+1}([k]) = AC$ , since once the cups in  $B$  are emptied from, the cups in  $B$  are not among the  $k$  fullest cups, so cups in  $C$  take their places among the  $k$  fullest cups.

Note that  $k - a \geq 0$  as  $a + b \leq k$ , and also  $|ABC| = k + b \leq n$ , because by definition the  $b$  cups in  $B$  must not be among the  $k$  fullest cups in state  $S_{t+1}$  so there are at least  $k + b$  cups. Note that  $a + b = \min(k, p_t)$ . We also have that  $A = I_t([a])$  and  $B = I_t(a + [b])$ , as every cup in  $A$  must have higher fill than all cups in  $B$  in order to remain above the cups in  $B$  after 1 unit of water is removed from all cups in  $AB$ .

We now establish the following claim, which we call the **interchangeability of cups**:

► **Claim 3.** There exists a cup state  $S'_t$  such that: (a)  $S'_t$  satisfies the invariants (1), (b)  $S'_t(r) = I_t(r)$  for all ranks  $r \in [n]$ , and (c) the filler can legally place water into cups in order to transform  $S'_t$  into  $I_t$ .

**Proof.** Fix  $r \in [n]$ . We will show that  $S_t$  can be transformed into a state  $S'_t$  by relabelling only cups with ranks in  $[r]$  such that (a)  $S'_t$  satisfies the invariants (1), (b)  $S'_t([r]) = I_t([r])$  and (c) the filler can legally place water into cups in order to transform  $S'_t$  into  $I_t$ .



## 16:6 The Variable-Processor Cup Game

Say there are cups  $x, y$  with  $x \in S_t([r]) \setminus I_t([r]), y \in I_t([r]) \setminus S_t([r])$ . Let the fills of cups  $x, y$  at state  $S_t$  be  $f_x, f_y$ ; note that

$$f_x > f_y. \quad (2)$$

Let the amount of fill that the filler adds to these cups be  $\Delta_x, \Delta_y \in [0, 1]$ ; note that

$$f_x + \Delta_x < f_y + \Delta_y. \quad (3)$$

Define a new state  $S'_t$  where cup  $x$  has fill  $f_y$  and cup  $y$  has fill  $f_x$ . Note that the filler can transform state  $S'_t$  into state  $I_t$  by placing water into cups as before, except changing the amount of water placed into cups  $x$  and  $y$  to be  $f_x - f_y + \Delta_x$  and  $f_y - f_x + \Delta_y$ , respectively.

In order to verify that the transformation from  $S'_t$  to  $I_t$  is a valid step for the filler, one must check three conditions. First, the amount of water placed by the filler is unchanged: this is because  $(f_x - f_y + \Delta_x) + (f_y - f_x + \Delta_y) = \Delta_x + \Delta_y$ . Second, the fills placed in cups  $x$  and  $y$  are at most 1: this is because  $f_x - f_y + \Delta_x < \Delta_y \leq 1$  (by (3)) and  $f_y - f_x + \Delta_x < \Delta_x \leq 1$  (by (2)). Third, the fills placed in cups  $x$  and  $y$  are non-negative: this is because  $f_x - f_y + \Delta_x > \Delta_x \geq 0$  (by (2)) and  $f_y - f_x + \Delta_y > \Delta_y \geq 0$  (by (3)).

We can repeatedly apply this process to swap each cup in  $I_t([r]) \setminus S_t([r])$  into being in  $S'_t([r])$ . At the end of this process we will have some state  $S'_t$  for which  $S'_t([r]) = I_t([r])$ . Note that  $S'_t$  is simply a relabeling of  $S_t$ , hence it must satisfy the same invariants (1) satisfied by  $S_t$ . Further,  $S'_t$  can be transformed into  $I_t$  by a valid filling step.

Now we repeatedly apply this process, in descending order of ranks. In particular, we have the following process: create a sequence of states by starting with  $S_t^{n-1}$ , and to get to state  $S_t^r$  from state  $S_t^{r+1}$  apply the process described above. Note that  $S_t^{n-1}$  satisfies  $S_t^{n-1}([n-1]) = I_t([n-1])$  and thus also  $S_t^{n-1}(n) = I_t(n)$ . If  $S_t^{r+1}$  satisfies  $S_t^{r+1}(r') = I_t(r')$  for all  $r' > r+1$  then  $S_t^r$  satisfies  $S_t^r(r') = I_t(r')$  for all  $r' > r$ , because the transition from  $S_t^{r+1}$  to  $S_t^r$  has not changed the labels of any cups with ranks in  $(r+1, n]$ , but the transition does enforce  $S_t^r([r]) = I_t([r])$ , and consequently  $S_t^r(r+1) = I_t(r+1)$ . We continue with the sequential process until arriving at state  $S_t^1$  in which we have  $S_t^1(r) = I_t(r)$  for all  $r$ . Throughout the process each  $S_t^r$  has satisfied the invariants (1), so  $S_t^1$  satisfies the invariants (1). Further, throughout the process from each  $S_t^r$  it is possible to legally place water into cups in order to transform  $S_t^r$  into  $I_t$ .

Hence  $S_t^1$  satisfies all the properties desired, and the proof of Claim 3 is complete.  $\triangleleft$

Claim 3 tells us that we may assume without loss of generality that  $S_t(r) = I_t(r)$  for each rank  $r \in [n]$ . We will make this assumption for the rest of the proof.

In order to complete the proof of the theorem, we break it into three cases.

$\triangleright$  **Claim 4.** If some cup in  $A$  zeroes out in round  $t$ , then the invariant  $\mu_{S_{t+1}}(S_{t+1}([k])) \leq 2n-k$  holds.

**Proof.** Say a cup in  $A$  zeroes out in step  $t$ . Of course

$$m_{S_{t+1}}(I_t([a-1])) \leq (a-1)(2n-(a-1))$$

because the  $a-1$  fullest cups must have satisfied the invariant (with  $k = a-1$ ) on round  $t$ . Moreover, because  $\text{fill}_{S_{t+1}}(I_{t+1}(a)) = 0$

$$m_{S_{t+1}}(I_t([a])) = m_{S_{t+1}}(I_t([a-1])).$$

Combining the above equations, we get that

$$m_{S_{t+1}}(A) \leq (a-1)(2n-(a-1)).$$

Furthermore, the fill of all cups in  $C$  must be at most 1 at state  $I_t$  to be less than the fill of the cup in  $A$  that zeroed out. Thus,

$$\begin{aligned}
 m_{S_{t+1}}(S_{t+1}([k])) &= m_{S_{t+1}}(AC) \\
 &\leq (a-1)(2n-(a-1)) + k - a \\
 &= a(2n-a) + a - 2n + a - 1 + k - a \\
 &= a(2n-a) + (k-n) + (a-n) - 1 \\
 &< a(2n-a)
 \end{aligned}$$

as desired. As  $k$  increases from 1 to  $n$ ,  $k(2n-k)$  strictly increases (it is a quadratic in  $k$  that achieves its maximum value at  $k = n$ ). Thus  $a(2n-a) \leq k(2n-k)$  because  $a \leq k$ . Therefore,

$$m_{S_{t+1}}(S_{t+1}([k])) \leq k(2n-k). \quad \triangleleft$$

▷ **Claim 5.** If no cups in  $A$  zero out in round  $t$  and  $b = 0$ , then the invariant  $\mu_{S_{t+1}}(S_{t+1}([k])) \leq 2n - k$  holds.

Proof. If  $b = 0$ , then  $S_{t+1}([k]) = S_t([k])$ . During round  $t$  the emptier removes  $a$  units of fill from the cups in  $S_t([k])$ , specifically the cups in  $A$ . The filler cannot have added more than  $k$  fill to these cups, because it can add at most 1 fill to any given cup. Also, the filler cannot have added more than  $p_t$  fill to the cups because this is the total amount of fill that the filler is allowed to add. Hence the filler adds at most  $\min(p_t, k) = a + b = a + 0 = a$  fill to these cups. Thus the invariant holds:

$$m_{S_{t+1}}(S_{t+1}([k])) \leq m_{S_t}(S_t([k])) + a - a \leq k(2n-k). \quad \triangleleft$$

The remaining case, in which no cups in  $A$  zero out and  $b > 0$  is the most technically interesting.

▷ **Claim 6.** If no cups in  $A$  zero out on round  $t$  and  $b > 0$ , then the invariant  $\mu_{S_{t+1}}(S_{t+1}([k])) \leq 2n - k$  holds.

Proof. Because  $b > 0$  and  $a + b \leq k$  we have that  $a < k$ , and  $c = k - a > 0$ . Recall that  $S_{t+1}([k]) = AC$ , so the mass of the  $k$  fullest cups at  $S_{t+1}$  is the mass of  $AC$  at  $S_t$  plus any water added to cups in  $AC$  by the filler, minus any water removed from cups in  $AC$  by the emptier. The emptier removes exactly  $a$  units of water from  $AC$ . The filler adds no more than  $p_t$  units of water to  $AC$  (because the filler adds at most  $p_t$  total units of water per round) and the filler also adds no more than  $k = |AC|$  units of water to  $AC$  (because the filler adds at most 1 unit of water to each of the  $k$  cups in  $AC$ ). Thus, the filler adds no more than  $a + b = \min(p_t, k)$  units of water to  $AC$ . Combining these observations we have:

$$m_{S_{t+1}}(S_{t+1}([k])) \leq m_{S_t}(AC) + b. \quad (4)$$

The key insight necessary to bound this is to notice that larger values for  $m_{S_t}(A)$  correspond to smaller values for  $m_{S_t}(C)$  because of the invariants; the higher fill in  $A$  **pushes down** the fill that  $C$  can have. By capturing the pushing-down relationship combinatorially we will achieve the desired inequality.

We can upper bound  $m_{S_t}(C)$  by

$$\begin{aligned}
 m_{S_t}(C) &\leq \frac{c}{b+c} m_{S_t}(BC) \\
 &= \frac{c}{b+c} (m_{S_t}(ABC) - m_{S_t}(A))
 \end{aligned}$$

## 16:8 The Variable-Processor Cup Game

because  $\mu_{S_t}(C) \leq \mu_{S_t}(B)$  without loss of generality by the interchangeability of cups. Thus we have

$$m_{S_t}(AC) \leq m_{S_t}(A) + \frac{c}{b+c} m_{S_t}(BC) \quad (5)$$

$$= \frac{c}{b+c} m_{S_t}(ABC) + \frac{b}{b+c} m_{S_t}(A). \quad (6)$$

Note that the expression in (6) is monotonically increasing in both  $\mu_{S_t}(ABC)$  and  $\mu_{S_t}(A)$ . Thus, by numerically replacing both average fills with their extremal values,  $2n - |ABC|$  and  $2n - |A|$ . At this point the claim can be verified by straightforward (but quite messy) algebra (and by combining (4) with (6)). We instead give a more intuitive argument, in which we examine the right side of (5) combinatorially.

Consider a new configuration of fills  $F$  achieved by starting with state  $S_t$ , and moving water from  $BC$  into  $A$  until  $\mu_F(A) = 2n - |A|$ .<sup>2</sup> This transformation increases (strictly increases if and only if we move a non-zero amount of water) the right side of (5). In particular, if mass  $\Delta \geq 0$  fill is moved from  $BC$  to  $A$ , then the right side of (5) increases by  $\frac{b}{b+c} \Delta \geq 0$ . Note that the fact that moving water from  $BC$  into  $A$  increases the right side of (5) formally captures the way the system of invariants being proven forces a tradeoff between the fill in  $A$  and the fill in  $BC$  – that is, higher fill in  $A$  pushes down the fill that  $BC$  (and consequently  $C$ ) can have.

Since  $\mu_F(A)$  is above  $\mu_F(ABC)$ , the greater than average fill of  $A$  must be counter-balanced by the lower than average fill of  $BC$ . In particular we must have

$$(\mu_F(A) - \mu_F(ABC))|A| = (\mu_F(ABC) - \mu_F(BC))|BC|.$$

Note that

$$\begin{aligned} & \mu_F(A) - \mu_F(ABC) \\ &= (2n - |A|) - \mu_F(ABC) \\ &\geq (2n - |A|) - (2n - |ABC|) \\ &= |BC|. \end{aligned}$$

Hence we must have

$$\mu_F(ABC) - \mu_F(BC) \geq |A|.$$

Thus

$$\mu_F(BC) \leq \mu_F(ABC) - |A| \leq 2n - |ABC| - |A|. \quad (7)$$

Combining (5) with the fact that the transformation from  $S_t$  to  $F$  only increases the right side of (5), along with (7), we have the following bound:

$$\begin{aligned} m_{S_t}(AC) &\leq m_F(A) + c\mu_F(BC) \\ &\leq a(2n - a) + c(2n - |ABC| - a) \\ &\leq (a + c)(2n - a) - c(a + c + b) \\ &\leq (a + c)(2n - a - c) - cb. \end{aligned} \quad (8)$$

---

<sup>2</sup> Note that whether or not  $F$  satisfies the invariants is irrelevant.

By (4) and (8), we have that

$$\begin{aligned}
 m_{S_{t+1}}(S_{t+1}([k])) &\leq m_{S_t}(AC) + b \\
 &\leq (a + c)(2n - a - c) - cb + b \\
 &= k(2n - k) - cb + b \\
 &\leq k(2n - k),
 \end{aligned}$$

where the final inequality uses the fact that  $c \geq 1$ . This completes the proof of the claim.  $\triangleleft$

We have shown the invariant holds for arbitrary  $k$ , so given that the invariants all hold at state  $S_t$  they also must all hold at state  $S_{t+1}$ . Thus, by induction we have the invariant for all rounds  $t \in \mathbb{N}$ .  $\blacktriangleleft$

## 4 Technical overview of lower bounds

The rest of the paper is devoted to the construction of filler strategies that match (or nearly match) the upper bound in Section 3.

This section gives a technical overview of the main technical ideas used in the filler constructions. Full versions of the constructions are given in Section 5 (for adaptive fillers) and in the extended version of this paper (for oblivious fillers).

### 4.1 Adaptive Lower Bound

In Section 5 we provide an adaptive filling strategy that achieves backlog  $\Omega(n^{1-\epsilon})$ ; in this subsection we sketch the proof of the result.

First we note that there is a trivial algorithm, that we call **trivalg**, for achieving backlog at least  $1/2$  on at least 2 cups in time  $O(1)$ .

The essential ingredient in our lower-bound construction is what we call the **Amplification Lemma**. The lemma takes as input a filling strategy that achieves some backlog curve  $f$  (i.e., on  $n$  cups, the strategy achieves backlog  $f(n)$ ), and outputs a new filling strategy that achieves a new amplified backlog curve  $f'$ .

► **Lemma 7** (Lemma 12). *Let  $\text{alg}(f)$  be a filling strategy that achieves backlog  $f(n)$  on  $n$  cups (in the negative-fill cup game). There exists a filling strategy  $\text{alg}(f')$ , the **amplification** of  $\text{alg}(f)$ , that achieves backlog at least*

$$f'(n) \geq (1 - \delta)f(\lfloor (1 - \delta)n \rfloor) + f(\lceil \delta n \rceil).$$

**Proof Sketch.** The filler designates an **anchor set**  $A$  of size  $\lceil \delta n \rceil$  and a **non-anchor set**  $B$  of size  $\lfloor (1 - \delta)n \rfloor$ .

The filler's strategy begins with  $M$  phases, for some parameter  $M$  to be determined later. In each phase, the filler applies  $\text{alg}(f)$  to the non-anchor set  $B$ , while simultaneously placing 1 unit of water into each cup of  $A$  on each step. If there is ever a step during the phase in which the emptier does not remove water from every cup in  $A$ , then the phase is said to be **emptier neglected**. On the other hand, if a phase is not emptier neglected, then at the end of the phase, the filler swaps the cup in  $B$  with largest fill with the cup in  $A$  whose fill is smallest.

After the  $M$  phases are complete, the filler then recursively applies  $\text{alg}(f)$  to the cups  $A$ . This completes the filling strategy  $\text{alg}(f')$ .

The key to analyzing  $\text{alg}(f')$  is to show that, at the end of the  $M$ -th phase, the average fill of the cups  $A$  satisfies  $\mu(A) \geq (1 - \delta)f(|B|)$ . This, in turn, means that the recursive application of  $\text{alg}(f)$  to  $A$  will achieve backlog  $(1 - \delta)f(|B|) + f(|A|)$ , as desired.

Now let us reason about  $\mu(A)$ . If a phase is emptier neglected, then the total amount of water placed into  $A$  during the phase is at least 1 greater than the total amount of water removed. Hence  $\mu(A)$  increases by at least  $1/|A|$ . On the other hand, if a phase is not emptier neglected, then  $\text{alg}(f)$  will successfully achieve backlog  $\mu(B) + f(|B|)$  on the cups  $B$  during the phase. At the end of the phase, the filler will then swap a cup from  $B$  with large fill with a cup from  $A$ . Thus, in each phase, we either have that  $\mu(A)$  increases by  $1/|A|$ , or that a cup with large fill gets swapped into  $A$ . After sufficiently many phases, one can show that  $\mu(A)$  is guaranteed to become at least  $(1 - \delta)f(|B|) + f(|A|)$ . ◀

We use the Amplification Lemma to give two lower bounds on backlog: one with reasonable running time, the other with slightly better backlog.

► **Theorem 8** (Theorem 13). *There is an adaptive filling strategy for achieving backlog  $\Omega(n^{1-\varepsilon})$  for constant  $\varepsilon \in (0, 1/2)$  in running time  $2^{O(\log^2 n)}$ .*

**Proof Sketch.** We construct a sequence of filling strategies with  $\text{alg}(f_{i+1})$  the amplification of  $\text{alg}(f_i)$  using  $\delta = \Theta(1)$  determined as a function of  $\varepsilon$ , and  $\text{alg}(f_0) = \text{trivalg}$ . Choosing  $\delta$  appropriately as a function of  $\varepsilon$ , and letting  $c$  be some (small) positive constant, we show by induction on  $i$  that, for all  $k \leq 2^{ci}$ ,  $\text{alg}(f_i)$  achieves backlog  $\Omega(k^{1-\varepsilon})$  on  $k$  cups in running time  $2^{O(\log^2 k)}$ . Taking  $i = \Theta(\log n)$  completes the proof. ◀

► **Theorem 9** (Theorem 15). *There is an adaptive filling strategy for achieving backlog  $\Omega(n)$  in running time  $O(n!)$ .*

**Proof Sketch.** We construct a sequence of filling strategies with  $\text{alg}(f_{i+1})$  the amplification of  $\text{alg}(f_i)$  using  $\delta = 1/(i+1)$ , and  $\text{alg}(f_0)$  a filling strategy for achieving backlog 1 on  $O(1)$  cups in  $O(1)$  time (this is a slight modification of  $\text{trivalg}$ ). We show by induction that  $\text{alg}(f_{\Theta(n)})$  achieves backlog  $\Omega(n)$  in running time  $O(n!)$ . ◀

## 4.2 Oblivious Lower Bound

We now consider what happens if the filler is an **oblivious** adversary, meaning that the filler cannot see what the emptier does at each step. The emptier, in turn, is permitted to use randomization in order to make its behavior unpredictable to the filler. We focus on randomized emptying algorithms that satisfy the so-called  **$\Delta$ -greedy-like** property: the emptier never empties from a cup  $c$  over another cup  $c'$  whose fill is more than  $\Delta$  greater than the fill of  $c$ .

The next theorem gives an oblivious filling strategy that achieves backlog  $\Omega(n^{1-\varepsilon})$  against any  $\Delta$ -greedy-like emptier for any  $\Delta \in \Omega(1)$  (or, more precisely, any  $\Delta \leq \frac{1}{128} \log \log \log n$ ).

► **Theorem 10.** *There is an oblivious filling strategy for the variable-processor cup game on  $N$  cups that achieves backlog at least  $\Omega(N^{1-\varepsilon})$  for any constant  $\varepsilon > 0$  in running time  $2^{\text{polylog}(N)}$  with probability at least  $1 - 2^{-\text{polylog}(N)}$  against a  $\Delta$ -greedy-like emptier with  $\Delta \leq \frac{1}{128} \log \log \log N$ .*

Note that Theorem 10 uses  $N$  for the number of cups rather than using  $n$ . When describing the recursive strategy that the filler uses, we will use  $N$  to denote the true total number of cups, and  $n$  to denote the number of cups within the recursive subproblem currently being discussed.

The filling strategy used in Theorem 10 is closely related the adaptive filling strategy described in Section 4.1. The fact that the filling strategy must now be *oblivious*, however, introduces several new technical obstacles.

### Problem 1: Distinguishing between neglected and non-neglected phases

Recall that the Amplification Lemma in Section 4.1 proceeds in phases, where the filler behaves differently at the end of each phase depending on whether or not the emptier ever neglected the anchor set  $A$  during that phase. If the filler is oblivious, however, then it cannot detect which phases are neglected.

To solve this problem, the first thing to notice is that the *total* number of times that the emptier can neglect the anchor set (within a given recursive subproblem of the Amplification Lemma) is, without loss of generality, at most  $N^2$ . Indeed, if the emptier neglects the anchor set more than  $N^2$  times, then the total amount of water in cups  $A$  will be at least  $N^2$ . Since the amount of water in the system as a whole is non-decreasing, there will subsequently always be at least one cup in the system with fill  $N$  or larger, and thus the filler's strategy trivially achieves backlog  $N$ .

Assume that there are at most  $N^2$  phases that the emptier neglects. The filler does not know which phases these are, and the filler does not wish to ever move a cup from the non-anchor set to the anchor set during a phase that the emptier neglected (since, during such a phase, there is no guarantee on the amount of water in the cup from  $B$ ). To solve this problem, we increase the total number of phases in the Amplification Lemma to be some very large number  $M = 2^{\text{polylog } N}$ , and we have the filler select  $|A|$  random phases at the end of which to move a cup from the non-anchor set to the anchor set. With high probability, none of the  $|A|$  phases that the filler selects are neglected by the emptier.

### Problem 2: Handling the probability of failure

Because the filler is now oblivious (and the emptier is randomized) the guarantee offered by the filling strategy is necessarily probabilistic. This makes the Amplification Lemma somewhat more difficult, since each application of  $\text{alg}(f)$  now has some probability of failure.

We ensure that the applications of  $\text{alg}(f)$  succeed with such high probability that we can ignore the possibility of any of them failing on phases when we need them to succeed. This necessitates making sure that the base-case construction  $\text{alg}(f_0)$  succeeds with very high probability.

Fortunately, we can take a base-case construction  $\text{alg}_0$  that succeeds with only constant (or even sub-constant) probability, and perform an Amplification-Lemma-like construction in order to obtain a new filling strategy  $\text{alg}_1$  that achieves slightly *smaller* backlog, but that has a very high probability of succeeding.

To construct  $\text{alg}_1$ , we begin by performing the Amplification-Lemma construction on  $\text{alg}_0$ , but without recursing after the final phase. Even though many of the applications of  $\text{alg}_0$  may fail, with high probability at least one application succeeds. This results in some cup  $c_*$  in  $A$  having high fill. Unfortunately, the filler does not know which cup has high fill, so it cannot simply take  $c_*$ . What the filler can do, however, is select some cup  $c$ , decrease the number of processors to 1, and then spend a large number of steps simply placing 1 unit of water into cup  $c$  in each step. By the  $\Delta$ -greediness of the emptier, the emptier is guaranteed to focus on emptying from cup  $c_*$  (rather than cup  $c$ ) until  $c$  attains large fill. This allows for the filler to obtain a cup  $c$  that the filler *knows* contains a large amount of water (with high probability). We use this approach to construct a base-case filling strategy  $\text{alg}(f'_0)$  that succeeds with high probability.

**Problem 3: Non-flat starting states**

The next problem that we encounter is that, at the beginning of any given phase, the cups in the non-anchor set may not start off with equal heights. Instead, some cups may contain very large amounts of water while others contain very small (and even negative) amounts of water<sup>3</sup>. This is not a problem for an adaptive filler, since the filler knows which cups contain small/large amounts of water, but it is a problem for an oblivious filler.

To avoid the scenario in which the cups in  $B$  are highly unequal, we begin each phase by first performing a **flattening construction** on the cups  $B$ , which causes the cups in  $B$  to all have roughly equal fills (up to  $\pm O(\Delta)$ ). The flattening construction uses the fact that the emptier is  $\Delta$ -greedy-like to ensure that cups which are overly full get flattened out by the emptier.

**Putting the pieces together**

By combining the ideas above, as well as handling other issues that arise (e.g., one must be careful to ensure that the average fills of  $A$  and  $B$  do not drift apart in unpredictable ways), one can prove Theorem 10.

**5 Adaptive Filler Lower Bound**

In this section we give a  $2^{\text{polylog } n}$ -time filling strategy that achieves backlog  $n^{1-\varepsilon}$  for any positive constant  $\varepsilon$ . We also give a  $O(n!)$ -time filling strategy that achieves backlog  $\Omega(n)$ . These results formalize the ideas described in Section 4.1.

We begin with a trivial filling strategy that we call **trivalg** that gives backlog at least  $1/2$  when applied to at least 2 cups.

► **Proposition 11.** *Consider an instance of the negative-fill 1-processor cup game on  $n$  cups, and let the cups start in any state with average fill is 0. If  $n \geq 2$ , there is an  $O(1)$ -step adaptive filling strategy **trivalg** that achieves backlog at least  $1/2$ . If  $n = 1$ , **trivalg** achieves backlog 0 in running time 0.*

**Proof.** If  $n = 1$ , **trivalg** does nothing and achieves backlog 0; for the rest of the proof we consider the case  $n \geq 2$ .

Let  $a$  and  $b$  be the fullest and second fullest cups in the in the starting configuration, and let their initial fills be  $\text{fill}(a) = \alpha, \text{fill}(b) = \beta$ . If  $\alpha \geq 1/2$  the filler need not do anything, the desired backlog is already achieved. Otherwise, if  $\alpha \in [0, 1/2]$ , the filler places  $1/2 - \alpha$  fill into  $a$  and  $1/2 + \alpha$  fill into  $b$  (which is possible as both fills are in  $[0, 1]$ , and they sum to 1). Since  $\alpha + \beta \geq 0$  we have  $\beta \geq -\alpha$ . Clearly  $a$  and  $b$  now both have fill at least  $1/2$ . The emptier cannot empty from both  $a$  and  $b$  as  $p = 1$ , so even after the emptier empties from a cup we still have backlog  $1/2$ , as desired. ◀

Next we prove the **Amplification Lemma**, which takes as input a filling strategy  $\text{alg}(f)$  and outputs a new filling strategy  $\text{alg}(f')$  that we call the **amplification** of  $\text{alg}(f)$ .  $\text{alg}(f')$  is able to achieve higher fill than  $\text{alg}(f)$ ; in particular, we will show that by starting with a filling strategy  $\text{alg}(f_0)$  for achieving constant backlog and then forming a sufficiently long sequence of filling strategies  $\text{alg}(f_0), \text{alg}(f_1), \dots, \text{alg}(f_{i_*})$  with  $\text{alg}(f_{i+1})$  the amplification of  $\text{alg}(f_i)$ , we get a filling strategy for achieving  $\text{poly}(n)$  backlog.

<sup>3</sup> Recall that, in order for our lower-bound construction to be able to call itself recursively, we must analyze the construction in the negative-fill version of the variable-processor cup game.



► **Lemma 12** (Adaptive Amplification Lemma). *Let  $\delta \in (0, 1/2]$  be a parameter. Let  $\text{alg}(f)$  be an adaptive filling strategy that achieves backlog  $f(n) < n$  in the negative-fill variable-processor cup game on  $n$  cups in running time  $T(n)$  starting from any initial cup state where the average fill is 0.*

*Then there exists an adaptive filling strategy  $\text{alg}(f')$  that achieves backlog  $f'(n)$  satisfying*

$$f'(n) \geq (1 - \delta)f(\lfloor (1 - \delta)n \rfloor) + f(\lceil \delta n \rceil)$$

*and  $f'(n) \geq f(n)$  in the negative-fill variable-processor cup game on  $n$  cups in running time*

$$T'(n) \leq n \lceil \delta n \rceil \cdot T(\lfloor (1 - \delta)n \rfloor) + T(\lceil \delta n \rceil)$$

*starting from any initial cup state where the average fill is 0.*

**Proof.** Let  $n_A = \lceil \delta n \rceil$ ,  $n_B = n - n_A = \lfloor (1 - \delta)n \rfloor$ .

The filler defaults to using  $\text{alg}(f)$  if

$$f(n) \geq (1 - \delta)f(n_B) + f(n_A).$$

In this case using  $\text{alg}(f)$  achieves the desired backlog in the desired running time. In the rest of the proof, we describe our strategy for the case where we cannot simply use  $\text{alg}(f)$  to achieve the desired backlog.

Let  $A$ , the **anchor set**, be initialized to consist of the  $n_A$  fullest cups, and let  $B$  the **non-anchor set** be initialized to consist of the rest of the cups (so  $|B| = n_B$ ). Let  $h = (1 - \delta)f(n_B)$ .

The filler's strategy can be summarized as follows:

**Step 1:** Make  $\mu(A) \geq h$  by using  $\text{alg}(f)$  repeatedly on  $B$  to achieve cups with fill at least  $\mu(B) + f(n_B)$  in  $B$  and then swapping these into  $A$ . While doing this the filler always places 1 unit of fill in each anchor cup.

**Step 2:** Use  $\text{alg}(f)$  once on  $A$  to obtain some cup with fill  $\mu(A) + f(n_A)$ .

Note that in order to use  $\text{alg}(f)$  on subsets of the cups the filler will need to vary  $p$ .

We now describe how to achieve Step 1, which is complicated by the fact that the emptier may attempt to prevent the filler from achieving high fill in a cup in  $B$ .

The filling strategy always places 1 unit of water in each anchor cup. This ensures that no cups in the anchor set ever have their fill decrease. If the emptier wishes to keep the average fill of the anchor cups from increasing, then emptier must empty from every anchor cup on each step. If the emptier fails to do this on a given round, then we say that the emptier has **neglected** the anchor cups.

We say that the filler **applies**  $\text{alg}(f)$  to  $B$  if it follows the filling strategy  $\text{alg}(f)$  on  $B$  while placing 1 unit of water in each anchor cup. An application of  $\text{alg}(f)$  to  $B$  is said to be **successful** if  $A$  is never neglected during the application of  $\text{alg}(f)$  to  $B$ . The filler uses a series of phases that we call **swapping-processes** to achieve the desired average fill in  $A$ . In a swapping-process, the filler repeatedly applies  $\text{alg}(f)$  to  $B$  until a successful application occurs, and then takes the cup generated by  $\text{alg}(f)$  within  $B$  on this successful application with fill at least  $\mu(B) + f(|B|)$  and swaps it with the least full cup in  $A$  so long as the swap increases  $\mu(A)$ . If the average fill in  $A$  ever reaches  $h$ , then the algorithm immediately halts (even if it is in the middle of a swapping-process) and is complete.

We give pseudocode for the filling strategy in Algorithm 1.

## 16:14 The Variable-Processor Cup Game

■ **Algorithm 1** Adaptive Amplification (Step 1).

**Input:**  $\text{alg}(f), \delta$ , set of  $n$  cups

**Output:** Guarantees that  $\mu(A) \geq h$

$A \leftarrow n_A$  fullest cups,  $B \leftarrow$  rest of the cups

Always place 1 fill in each cup in  $A$

**while**  $\mu(A) < h$  **do**

▷ Swapping-Processes

Immediately **exit** this loop if ever  $\mu(A) \geq h$

successful  $\leftarrow$  false

**while** not successful **do**

Apply  $\text{alg}(f)$  to  $B$ ,  $\text{alg}(f)$  gives cup  $c$

**if**  $\text{fill}(c) \geq h$  **then**

successful  $\leftarrow$  true

Swap  $c$  with least full cup in  $A$

Note that

$$\mu(A) \cdot |A| + \mu(B) \cdot |B| = \mu(AB) \geq 0,$$

as  $\mu(AB)$  starts as 0, but could become positive if the emptier skips emptyings. Thus we have

$$\mu(A) \geq -\mu(B) \cdot \frac{\lfloor (1-\delta)n \rfloor}{\lceil \delta n \rceil} \geq -\frac{1-\delta}{\delta} \mu(B).$$

Thus, if at any point  $B$  has average fill lower than  $-h \cdot \delta / (1-\delta)$ , then  $A$  has average fill at least  $h$ , so the algorithm is finished. Thus we can assume in our analysis that

$$\mu(B) \geq -h \cdot \delta / (1-\delta). \quad (9)$$

We will now show that the filler applies  $\text{alg}(f)$  to  $B$  at most  $hn_A$  total times. Each time the emptier neglects the anchor set, the mass of the anchor set increases by 1. If the emptier neglects the anchor set  $hn_A$  times, then the average fill in the anchor set increases by  $h$ . Since  $\mu(A)$  started as at least 0, and since  $\mu(A)$  never decreases (note in particular that cups are only swapped into  $A$  if doing so will increase  $\mu(A)$ ), an increase of  $h$  in  $\mu(A)$  implies that  $\mu(A) \geq h$ , as desired.

Consider the fill of a cup  $c$  swapped into  $A$  at the end of a swapping-process. Cup  $c$ 's fill is at least  $\mu(B) + f(n_B)$ , which by (9) is at least

$$-h \cdot \frac{\delta}{1-\delta} + f(n_B) = (1-\delta)f(n_B) = h.$$

Thus the algorithm for Step 1 succeeds within  $|A|$  swapping-processes, since at the end of the  $|A|$ -th swapping process either every cup in  $A$  has fill at least  $h$ , or the algorithm halted before  $|A|$  swapping-processes because it already achieved  $\mu(A) \geq h$ .

After achieving  $\mu(A) \geq h$ , the filler performs Step 2, i.e. the filler applies  $\text{alg}(f)$  to  $A$ , and hence achieves a cup with fill at least

$$\mu(A) + f(|A|) \geq (1-\delta)f(n_B) + f(n_A),$$

as desired.

Now we analyze the running time of the filling strategy  $\text{alg}(f')$ . First, recall that in Step 1  $\text{alg}(f')$  calls  $\text{alg}(f)$  on  $B$ , which has size  $n_B$ , as many as  $hn_A$  times. Because we mandate that  $h < n$ , Step 1 contributes no more than  $(n \cdot n_A) \cdot T(n_B)$  to the running time. Step 2 requires applying  $\text{alg}(f)$  to  $A$ , which has size  $n_A$ , once, and hence contributes  $T(n_A)$  to the running time. Summing these we have

$$T'(n) \leq n \cdot n_A \cdot T(n_B) + T(n_A). \quad \blacktriangleleft$$

We next show that by recursively using the Amplification Lemma we can achieve backlog  $n^{1-\varepsilon}$ .

► **Theorem 13.** *There is an adaptive filling strategy for the variable-processor cup game on  $n$  cups that achieves backlog  $\Omega(n^{1-\varepsilon})$  for any constant  $\varepsilon > 0$  of our choice in running time  $2^{O(\log^2 n)}$ .*

**Proof.** Take constant  $\varepsilon \in (0, 1/2)$ . Let  $c, \delta$  be constants that will be chosen (later) as functions of  $\varepsilon$  satisfying  $c \in (0, 1), 0 < \delta \ll 1/2$ . We show how to achieve backlog at least  $cn^{1-\varepsilon} - 1$ .

Let  $\text{alg}(f_0) = \text{trivalg}$ , the algorithm given by Proposition 11; recall  $\text{trivalg}$  achieves backlog  $f_0(k) \geq 1/2$  for all  $k \geq 2$ , and  $f_0(1) = 0$ . Next, using the Amplification Lemma we recursively construct  $\text{alg}(f_{i+1})$  as the amplification of  $\text{alg}(f_i)$  for  $i \geq 0$ . Define a sequence  $g_i$  with

$$g_i = \begin{cases} \lceil 16/\delta \rceil, & i = 0, \\ \lfloor g_{i-1}/(1-\delta) \rfloor & i \geq 1. \end{cases}$$

We claim the following regarding this construction:

▷ **Claim 14.** For all  $i \geq 0$ ,

$$f_i(k) \geq ck^{1-\varepsilon} - 1 \quad \text{for all } k \in [g_i]. \quad (10)$$

**Proof.** We prove Claim 14 by induction on  $i$ . For  $i = 0$ , the base case, (10) can be made true by taking  $c$  sufficiently small; in particular, taking  $c < 1$  makes (10) hold for  $k = 1$ , and, as  $g_0 > 2$ , taking  $c$  small enough to make  $cg_0^{1-\varepsilon} - 1 \leq f_0(g_0) = 1/2$  makes (10) hold for  $k \in [2, g_0]$  by monotonicity of  $k \mapsto ck^{1-\varepsilon} - 1$ .

As our inductive hypothesis we assume (10) for  $f_i$ ; we aim to show that (10) holds for  $f_{i+1}$ . Note that, by design of  $g_i$ , if  $k \leq g_{i+1}$  then  $\lfloor k \cdot (1-\delta) \rfloor \leq g_i$ . Consider any  $k \in [g_{i+1}]$ . First we deal with the trivial case where  $k \leq g_0$ . In this case

$$f_{i+1}(k) \geq f_i(k) \geq \dots \geq f_0(k) \geq ck^{1-\varepsilon} - 1.$$

Now we consider the case where  $k \geq g_0$ . Since  $f_{i+1}$  is the amplification of  $f_i$  we have

$$f_{i+1}(k) \geq (1-\delta)f_i(\lfloor (1-\delta)k \rfloor) + f_i(\lceil \delta k \rceil).$$

By our inductive hypothesis, which applies as  $\lceil \delta k \rceil \leq g_i, \lfloor k \cdot (1-\delta) \rfloor \leq g_i$ , we have

$$f_{i+1}(k) \geq (1-\delta)(c \cdot \lfloor (1-\delta)k \rfloor^{1-\varepsilon} - 1) + c \lceil \delta k \rceil^{1-\varepsilon} - 1.$$

---

<sup>4</sup> Note that it is important here that  $\varepsilon$  and  $\delta$  are constants, that way  $c$  is also a constant.

## 16:16 The Variable-Processor Cup Game

Dropping the floor and ceiling, incurring a  $-1$  for dropping the floor, we have

$$f_{i+1}(k) \geq (1-\delta)(c \cdot ((1-\delta)k - 1)^{1-\varepsilon} - 1) + c(\delta k)^{1-\varepsilon} - 1.$$

Because  $(x-1)^{1-\varepsilon} \geq x^{1-\varepsilon} - 1$ , as  $x \mapsto x^{1-\varepsilon}$  is a sub-linear sub-additive function, we have

$$f_{i+1}(k) \geq (1-\delta)c \cdot (((1-\delta)k)^{1-\varepsilon} - 2) + c(\delta k)^{1-\varepsilon} - 1.$$

Moving the  $ck^{1-\varepsilon}$  to the front we have

$$f_{i+1}(k) \geq ck^{1-\varepsilon} \cdot \left( (1-\delta)^{2-\varepsilon} + \delta^{1-\varepsilon} - \frac{2(1-\delta)}{k^{1-\varepsilon}} \right) - 1.$$

Because  $(1-\delta)^{2-\varepsilon} \geq 1 - (2-\varepsilon)\delta$ , a fact called Bernoulli's Identity, we have

$$f_{i+1}(k) \geq ck^{1-\varepsilon} \cdot \left( 1 - (2-\varepsilon)\delta + \delta^{1-\varepsilon} - \frac{2(1-\delta)}{k^{1-\varepsilon}} \right) - 1.$$

Of course  $-2(1-\delta) \geq -2$ , so

$$f_{i+1}(k) \geq ck^{1-\varepsilon} \cdot \left( 1 - (2-\varepsilon)\delta + \delta^{1-\varepsilon} - \frac{2}{k^{1-\varepsilon}} \right) - 1.$$

Because

$$\frac{-2}{k^{1-\varepsilon}} \geq \frac{-2}{g_0^{1-\varepsilon}} \geq -2(\delta/16)^{1-\varepsilon} \geq -\delta^{1-\varepsilon}/2,$$

which follows from our choice of  $g_0 = \lceil 16/\delta \rceil$  and the restriction  $\varepsilon < 1/2$ , we have

$$f_{i+1}(k) \geq ck^{1-\varepsilon} \cdot (1 - (2-\varepsilon)\delta + \delta^{1-\varepsilon} - \delta^{1-\varepsilon}/2) - 1.$$

Finally, combining terms we have

$$f_{i+1}(k) \geq ck^{1-\varepsilon} \cdot (1 - (2-\varepsilon)\delta + \delta^{1-\varepsilon}/2) - 1.$$

Because  $\delta^{1-\varepsilon}$  dominates  $\delta$  for sufficiently small  $\delta$ , there is a choice of  $\delta = \Theta(1)$  such that

$$1 - (2-\varepsilon)\delta + \delta^{1-\varepsilon}/2 \geq 1.$$

Taking  $\delta$  to be this small we have,

$$f_{i+1}(k) \geq ck^{1-\varepsilon} - 1,$$

completing the proof. We remark that the choices of  $c, \delta$  are the same for every  $i$  in the inductive proof, and depend only on  $\varepsilon$ .  $\triangleleft$

To complete the proof, we will show that  $g_i$  grows exponentially in  $i$ . This implies that there exists  $i_* \leq O(\log n)$  such that  $g_{i_*} \geq n$ , and hence we have an algorithm  $\text{alg}(f_{i_*})$  that achieves backlog  $cn^{1-\varepsilon} - 1$  on  $n$  cups, as desired.

We lower bound the sequence  $g_i$  with another sequence  $g'_i$  defined as

$$g'_i = \begin{cases} 4/\delta, & i = 0 \\ g'_{i-1}/(1-\delta) - 1, & i > 0. \end{cases}$$

Solving this recurrence, we find

$$g'_i = \frac{4 - (1 - \delta)^2}{\delta} \frac{1}{(1 - \delta)^i} \geq \frac{1}{(1 - \delta)^i},$$

which clearly exhibits exponential growth. In particular, let  $i_* = \lceil \log_{1/(1-\delta)} n \rceil$ . Then,  $g_{i_*} \geq g'_{i_*} \geq n$ , as desired.

Let the running time of  $f_i(n)$  be  $T_i(n)$ . From the Amplification Lemma we have following recurrence bounding  $T_i(n)$ :

$$\begin{aligned} T_i(n) &\leq n \lceil \delta n \rceil \cdot T_{i-1}(\lfloor (1 - \delta)n \rfloor) + T_{i-1}(\lceil \delta n \rceil) \\ &\leq 2n^2 T_{i-1}(\lfloor (1 - \delta)n \rfloor). \end{aligned}$$

It follows that  $\text{alg}(f_{i_*})$ , recalling that  $i_* \leq O(\log n)$ , has running time

$$T_{i_*}(n) \leq (2n^2)^{O(\log n)} \leq 2^{O(\log^2 n)},$$

as desired. ◀

Now we provide a construction that can achieve backlog  $\Omega(n)$  in very long games. The construction can be interpreted as the same argument as in Theorem 13 but with an extremal setting of  $\delta$  to  $\Theta(1/n)^5$ .

► **Theorem 15.** *There is an adaptive filling strategy that achieves backlog  $\Omega(n)$  in time  $O(n!)$ .*

**Proof.** First we construct a slightly stronger version of  $\text{trivalg}$  that achieves backlog 1 on  $n \geq n_0 = 8$  cups, instead of just backlog  $1/2$ ; this simplifies the analysis.

▷ **Claim 16.** There is a filling algorithm  $\text{trivalg}_2$  that achieves backlog at least 1 on  $n_0 = 8$  cups.

**Proof.** Let  $\text{trivalg}_1$  be the amplification of  $\text{trivalg}$  using  $\delta = 1/2$ . On 4 cups  $\text{trivalg}_1$  achieves backlog at least  $(1/2)(1/2) + 1/2 = 3/4$ . Let  $\text{trivalg}_2$  be the amplification of  $\text{trivalg}_1$  using  $\delta = 1/2$ . On 8 cups  $\text{trivalg}_2$  achieves backlog at least  $(1/2)(3/4) + 3/4 \geq 1$ . ◁

Let  $\text{alg}(f_0) = \text{trivalg}_2$ ; we have  $f_0(k) \geq 1$  for all  $k \geq n_0$ . For  $i > 0$  we construct  $\text{alg}(f_i)$  as the amplification of  $\text{alg}(f_{i-1})$  using the Amplification Lemma with parameter  $\delta = 1/(i+1)$ .

We claim the following regarding this construction:

▷ **Claim 17.** For all  $i \geq 0$ ,

$$f_i((i+1)n_0) \geq \sum_{j=0}^i \left(1 - \frac{j}{i+1}\right). \quad (11)$$

**Proof.** We prove Claim 17 by induction on  $i$ . When  $i = 0$ , the base case, (11) becomes  $f_0(n_0) \geq 1$  which is true. Assuming (11) for  $f_{i-1}$ , we now show (11) holds for  $f_i$ . Because  $f_i$  is the amplification of  $f_{i-1}$  with  $\delta = 1/(i+1)$ , we have by the Amplification Lemma

$$f_i((i+1) \cdot n_0) \geq \left(1 - \frac{1}{i+1}\right) f_{i-1}(i \cdot n_0) + f_{i-1}(n_0).$$

<sup>5</sup> Or more precisely, setting  $\delta$  in each level of recursion to be  $\Theta(1/n)$ , where  $n$  is the subproblem size; note in particular that  $\delta$  changes between levels of recursion, which was not the case in the proof of Theorem 13.

## 16:18 The Variable-Processor Cup Game

Since  $f_{i-1}(n_0) \geq f_0(n_0) \geq 1$  we have

$$f_i((i+1) \cdot n_0) \geq \left(1 - \frac{1}{i+1}\right) f_{i-1}(i \cdot n_0) + 1.$$

Using the inductive hypothesis we have

$$f_i((i+1) \cdot n_0) \geq \left(1 - \frac{1}{i+1}\right) \sum_{j=0}^{i-1} \left(1 - \frac{j}{i}\right) + 1.$$

Note that

$$\left(1 - \frac{1}{i+1}\right) \cdot \left(1 - \frac{j}{i}\right) = \frac{i}{i+1} \cdot \frac{i-j}{i} = \frac{i-j}{i+1} = 1 - \frac{j+1}{i+1}.$$

Thus we have the desired bound:

$$f_i((i+1) \cdot n_0) \geq \sum_{j=1}^i \left(1 - \frac{j}{i+1}\right) + 1 = \sum_{j=0}^i \left(1 - \frac{j}{i+1}\right). \quad \triangleleft$$

Let  $i_* = \lfloor n/n_0 \rfloor - 1$ , which by design satisfies  $(i_* + 1)n_0 \leq n$ . By Claim 17 we have

$$f_{i_*}((i_* + 1) \cdot n_0) \geq \sum_{j=0}^{i_*} \left(1 - \frac{j}{i_* + 1}\right) = i_*/2 + 1.$$

As  $i_* = \Theta(n)$ , we have thus shown that  $\text{alg}(f_{i_*})$  can achieve backlog  $\Omega(n)$  on  $n$  cups.

Let  $T_i$  be the running time of  $\text{alg}(f_i)$ . The recurrence for the running time of  $f_{i_*}$  is

$$T_i(n) \leq n \cdot n_0 T_{i-1}(n - n_0) + O(1).$$

Clearly  $T_{i_*}(n) \leq O(n!)$ . ◀

---

### References

- 1 Micah Adler, Petra Berenbrink, Tom Friedetzky, Leslie Ann Goldberg, Paul Goldberg, and Mike Paterson. A proportionate fair scheduling rule with good worst-case performance. In *Proceedings of the Fifteenth Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, pages 101–108, 2003. doi:10.1145/777412.777430.
- 2 Amihoud Amir, Martin Farach, Ramana M. Idury, Johannes A. La Poutré, and Alejandro A. Schäffer. Improved dynamic dictionary matching. *Inf. Comput.*, 119(2):258–282, 1995. doi:10.1006/inco.1995.1090.
- 3 Amihoud Amir, Gianni Franceschini, Roberto Grossi, Tsvi Kopelowitz, Moshe Lewenstein, and Noa Lewenstein. Managing unbounded-length keys in comparison-driven data structures with applications to online indexing. *SIAM Journal on Computing*, 43(4):1396–1416, 2014.
- 4 Yossi Azar and Arik Litichevsky. Maximizing throughput in multi-queue switches. *Algorithmica*, 45(1):69–90, 2006.
- 5 Amotz Bar-Noy, Ari Freund, Shimon Landa, and Joseph (Seffi) Naor. Competitive on-line switching policies. In *Proceedings of the Thirteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 525–534, 2002. URL: <http://dl.acm.org/citation.cfm?id=545381.545452>.
- 6 Amotz Bar-Noy, Aviv Nisgav, and Boaz Patt-Shamir. Nearly optimal perfectly periodic schedules. *Distributed Computing*, 15(4):207–220, 2002.

- 7 S. K. Baruah, N. K. Cohen, C. G. Plaxton, and D. A. Varvel. Proportionate progress: A notion of fairness in resource allocation. *Algorithmica*, 15(6):600–625, June 1996. doi:10.1007/BF01940883.
- 8 Sanjoy K Baruah, Johannes E Gehrke, and C Greg Plaxton. Fast scheduling of periodic tasks on multiple resources. In *Proceedings of the 9th International Parallel Processing Symposium*, pages 280–288, 1995.
- 9 Michael Bender, Rathish Das, Martín Farach-Colton, Rob Johnson, and William Kuszmaul. Flushing without cascades. In *Proceedings of the Thirty-First Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2020.
- 10 Michael Bender, Martín Farach-Colton, and William Kuszmaul. Achieving optimal backlog in multi-processor cup games. In *Proceedings of the 51st Annual ACM Symposium on Theory of Computing (STOC)*, 2019.
- 11 Michael Bender and William Kuszmaul. Randomized cup game algorithms against strong adversaries. In *Proceedings of the Thirty-Second Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2021.
- 12 Michael A Bender, Rezaul A Chowdhury, Rathish Das, Rob Johnson, William Kuszmaul, Andrea Lincoln, Quanquan C Liu, Jayson Lynch, and Helen Xu. Closing the gap between cache-oblivious and cache-adaptive analysis. In *Proceedings of the 32nd ACM Symposium on Parallelism in Algorithms and Architectures*, pages 63–73, 2020.
- 13 Michael A Bender, Rezaul A Chowdhury, Rathish Das, Rob Johnson, William Kuszmaul, Andrea Lincoln, Quanquan C Liu, Jayson Lynch, and Helen Xu. Closing the gap between cache-oblivious and cache-adaptive analysis. In *Proceedings of the 32nd ACM Symposium on Parallelism in Algorithms and Architectures*, pages 63–73, 2020.
- 14 Michael A Bender, Roozbeh Ebrahimi, Jeremy T Fineman, Golnaz Ghasemiesfeh, Rob Johnson, and Samuel McCauley. Cache-adaptive algorithms. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 958–971. SIAM, 2014.
- 15 Peter Damaschke and Zhen Zhou. On queuing lengths in on-line switching. *Theoretical computer science*, 339(2-3):333–343, 2005.
- 16 Paul Dietz and Daniel Sleator. Two algorithms for maintaining order in a list. In *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing (STOC)*, pages 365–372, 1987. doi:10.1145/28395.28434.
- 17 Paul F. Dietz and Rajeev Raman. Persistence, amortization and randomization. In *Proceedings of the Second Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 78–88, 1991. URL: <http://dl.acm.org/citation.cfm?id=127787.127809>.
- 18 Johannes Fischer and Paweł Gawrychowski. Alphabet-dependent string searching with wexponential search trees. In *Annual Symposium on Combinatorial Pattern Matching (CPM)*, pages 160–171, 2015.
- 19 Rudolf Fleischer and Hisashi Koga. Balanced scheduling toward loss-free packet queuing and delay fairness. *Algorithmica*, 38(2):363–376, February 2004. doi:10.1007/s00453-003-1064-z.
- 20 H Richard Gail, G Grover, Roch Guérin, Sidney L Hantler, Zvi Rosberg, and Moshe Sidi. Buffer size requirements under longest queue first. *Performance Evaluation*, 18(2):133–140, 1993.
- 21 Leszek Gasieniec, Ralf Klasing, Christos Levcopoulos, Andrzej Lingas, Jie Min, and Tomasz Radzik. Bamboo garden trimming problem (perpetual maintenance of machines with different attendance urgency factors). In *International Conference on Current Trends in Theory and Practice of Informatics*, pages 229–240. Springer, 2017.
- 22 Michael H. Goldwasser. A survey of buffer management policies for packet switches. *SIGACT News*, 41(1):100–128, 2010. doi:10.1145/1753171.1753195.
- 23 Michael T Goodrich and Paweł Pszozna. Streamed graph drawing and the file maintenance problem. In *International Symposium on Graph Drawing*, pages 256–267. Springer, 2013.



- 24 Nan Guan and Wang Yi. Fixed-priority multiprocessor scheduling: Critical instant, response time and utilization bound. In *2012 IEEE 26th International Parallel and Distributed Processing Symposium Workshops & PhD Forum*, pages 2470–2473. IEEE, 2012.
- 25 Tsvi Kopelowitz. On-line indexing for general alphabets via predecessor queries on subsets of an ordered list. In *Proceedings of the 53rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 283–292, 2012.
- 26 William Kuszmaul. Achieving optimal backlog in the vanilla multi-processor cup game. In *Proceedings of the Thirty-First Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2020.
- 27 Andrea Lincoln, Quanquan C Liu, Jayson Lynch, and Helen Xu. Cache-adaptive exploration: Experimental results and scan-hiding for adaptivity. In *Proceedings of the 30th on Symposium on Parallelism in Algorithms and Architectures*, pages 213–222, 2018.
- 28 Ami Litman and Shiri Moran-Schein. On distributed smooth scheduling. In *Proceedings of the Seventeenth Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 76–85, 2005.
- 29 Ami Litman and Shiri Moran-Schein. Smooth scheduling under variable rates or the analog-digital confinement game. *Theor. Comp. Sys.*, 45(2):325–354, June 2009. doi:10.1007/s00224-008-9134-x.
- 30 Ami Litman and Shiri Moran-Schein. On centralized smooth scheduling. *Algorithmica*, 60(2):464–480, 2011.
- 31 Chung Laung Liu. Scheduling algorithms for multiprocessors in a hard real-time environment. *JPL Space Programs Summary*, 1969, 1969.
- 32 Chung Laung Liu and James W Layland. Scheduling algorithms for multiprogramming in a hard-real-time environment. *Journal of the ACM (JACM)*, 20(1):46–61, 1973.
- 33 Richard T Mills, Andreas Stathopoulos, and Dimitrios S Nikolopoulos. Adapting to memory pressure from within scientific applications on multiprogrammed cows. In *Proc. 8th International Parallel and Distributed Processing Symposium (IPDPS)*, page 71, 2004.
- 34 Mark Moir and Srikanth Ramamurthy. Pfair scheduling of fixed and migrating periodic tasks on multiple resources. In *Proceedings of the 20th IEEE Real-Time Systems Symposium*, pages 294–303, 1999.
- 35 Christian Worm Mortensen. Fully-dynamic two dimensional orthogonal range and line segment intersection reporting in logarithmic time. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 618–627, 2003.
- 36 Michael Rosenblum, Michel X Goemans, and Vahid Tarokh. Universal bounds on buffer size for packetizing fluid policies in input queued, crossbar switches. In *Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, volume 2, pages 1126–1134, 2004.

# Comparison Graphs: A Unified Method for Uniformity Testing

Uri Meir

Tel Aviv University, Israel

---

## Abstract

Distribution testing can be described as follows:  $q$  samples are being drawn from some unknown distribution  $P$  over a known domain  $[n]$ . After the sampling process, a decision must be made about whether  $P$  holds some property, or is far from it. The most studied problem in the field is arguably *uniformity testing*, where one needs to distinguish the case that  $P$  is uniform over  $[n]$  from the case that  $P$  is  $\epsilon$ -far from being uniform (in  $\ell_1$ ). It is known that for this task  $\Theta(\sqrt{n}/\epsilon^2)$  samples are necessary and sufficient. This problem was recently considered in various restricted models that pose, for example, communication or memory constraints. In more than one occasion, the known optimal solution boils down to counting collisions among the drawn samples (each two samples that have the same value add one to the count). This idea dates back to the first uniformity tester, and was coined the name “collision-based tester”.

In this paper, we introduce the notion of *comparison graphs* and use it to formally define a generalized collision-based tester. Roughly speaking, the edges of the graph indicate the tester which pairs of samples should be compared (that is, the original tester is induced by a clique, where all pairs are being compared). We prove a structural theorem that gives a sufficient condition for a comparison graph to induce a good uniformity tester. As an application, we develop a generic method to test uniformity, and devise nearly-optimal uniformity testers under various computational constraints. We improve and simplify a few known results, and introduce a new constrained model in which the method also produces an efficient tester.

The idea behind our method is to translate computational constraints of a certain model to ones on the comparison graph, which paves the way to finding a good graph: a set of comparisons allowed by the model that suffice to test for uniformity. We believe that in future consideration of uniformity testing in new models, our method can be used to obtain efficient testers with minimal effort.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Distributed algorithms; Theory of computation  $\rightarrow$  Streaming, sublinear and near linear time algorithms

**Keywords and phrases** Distribution Testing, Uniformity Testing, Distributed Algorithms, Streaming Algorithms, Comparison Graphs

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.17

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/2012.01882>.

**Acknowledgements** The author would like to thank Rotem Oshman for valuable counsel, Noam Mazor for very helpful discussions, and Ami Paz as well as the anonymous referees of ITCS 2021 for their useful comments.

## 1 Introduction

The field of property testing was initiated by [10, 25, 20] and concerns with fast probabilistic algorithms that use query access to some large structure (such as graphs, functions, distribution, etc.) in order to determine whether a specific instance belongs to a subclass of possible instances (e.g., connected graphs or monotone functions) or in some sense far from it. Specifically for distributions, as formulated in [9], we are given random samples from some unknown distribution, and we wish to decide with high probability (over the random samples) whether it has some property, or it is far from any distribution that does (typically



© Uri Meir;

licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 17; pp. 17:1–17:20

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

we use  $\ell_1$  as a distance measure). Properties of distributions were excessively studied through the years (see [18, 11] for excellent surveys). Until recently, the vast majority of results were limited to the classic setting, where a *single* processor is given an oracle access, and performs the testing procedure. The measure of complexity is based solely on the number of samples, as typically the running time is polynomial in this number.

However, distribution testing can be very useful in different frameworks as well. For example, suppose we have a sensor network taking some measurements that need to be combined in order to make a decision about the subject of these measurements, be it volcanic activity, seismic movements, or any form of radiation.

Another framework where testing is useful, is under constrained memory. One might try testing an object so big, that the number of samples needed is very large. In that scenario, even if one might endure a lengthy sampling process, storing all past samples at one given moment can be too costly. For example, imagine a large telescope collecting data on infrared radiation in a pursuit to discover new planets. These types of questions can be translated to a *streaming model*, where samples come as a stream, and one wish to store only a small amount of data while reliably test for a property of the underlying distribution.

When considering the relatively similar motivations for these cases, one might even wonder about a combination of the two models, where multiple sensors spread out in some area collect samples, each of which has bounded memory. They will then need to process all available data within their memory constraints, such that by the end of each time period - they are able to report their individual findings concisely to some data center that aggregates all information in order to make an important decision.

All of these questions are inherently multi-dimensional, in the sense that many incomparable resources are in play: the number of players, the number of samples, the memory space of each sensor, and even the amount of bits communicated. Here, we focus our efforts, for the most part, on minimizing the *sample complexity* (or amount of samples per player, when multiple players are involved), with other resources given as parameters.

This specification is well-motivated by the case where we have no shortage of data we can sample from, and we wish to understand how long a sampling process should take, as a parameter of the number of sensors we use and their computational strength. Throughout the text, we focus on the task of uniformity testing, a key problem in the field of distribution testing.

**Uniformity testing.** The most studied family of problems in distribution testing is arguably *identity testing*, where we want to test whether the input distribution  $P$  is equal to some fixed distribution  $p$ , or  $\epsilon$ -far from it (in  $\ell_1$ ), where  $\epsilon$  is the proximity parameter of the problem. In the heart of these problems stands the problem of *uniformity testing*, where  $p = U_\Omega$ . One evidence for the importance of uniformity testing was shown in [15] and made more robust in [19]: it is actually complete for identity testing, in the sense that testing identity to any fixed distribution  $p$  can be reduced to testing uniformity instead. On the other hand, uniformity testing is a specific case of other problems such as closeness testing and independence testing, so showing lower bounds for uniformity testing would imply lower bounds for these problems. The classic version of testing uniformity was settled for the case  $\epsilon = \Omega(n^{-1/4})$  in [24], showing that  $\Theta(\sqrt{n}/\epsilon^2)$  samples are in fact sufficient and necessary. [26] later showed this holds for all values of  $\epsilon$ .

**Collision-based testers.** The problem of uniformity testing was implicitly introduced in [21], as a way to test the expansion of a graph: when simulating multiple short random walks, and observing the distribution of the endpoint, one can connect a uniform distribution over these

endpoint to good expansion of the graph. To solve uniformity, the *collision-based* tester was introduced, where one simply counts the number of pairs of samples that have the same value. This tester was shown to have sample complexity of  $\Theta(\sqrt{n} \cdot \text{poly}(1/\epsilon))$ , which is turned out to be sub-optimal in terms of  $\epsilon$ . However, several years after the question was settled, it was shown in [14] that the original collision-based tester also achieves optimal sample complexity, using a finer analysis.

The idea behind the collision-based tester is rather straightforward: when comparing two samples from a distribution, the chance of both having the same value (also referred as *collision probability*) relates to the  $\ell_2$  norm of the distribution. It is a well-known fact that over a fixed set  $\Omega$ , the uniform distribution has the minimal  $\ell_2$  norm. It is also rather easy to show that any distribution that is somewhat far from uniform (in statistical distance), has a significantly larger  $\ell_2$  norm. This means that each comparison of two samples is an unbiased estimator of the collision probability (having the right expectation), but with very high variance. One would need to average over many such comparisons in order to reduce the variance.

**Other methods to test for uniformity.** Over the years, numerous methods to test for uniformity have been proposed. Some of them had different goals in mind, such as testing with very high confidence, or in a multiparty model where each player gets a single sample (sometimes even wishing to keep it private). These methods include counting unique element [24], modified  $\chi^2$  test [26], using the empirical distance to uniformity [13], randomly hashing samples to a smaller domain [6] and more. In both [12, 16], testers that aim to overcome different constraints relied strongly on collision counting<sup>1</sup>. Considering it is also optimal in the classic setting, this makes collision-based testing a prime candidate for a more generic method to test uniformity, and hopefully adjust itself to different models easily.

**Comparison graphs.** The original version of the collision-based tester takes a set of samples, and use comparisons between *all pairs of samples*. This is well-suited for the classic model, where one processor sees all the samples, and can easily perform all comparisons.

However, in more constrained models, this simple task is inherently impossible. For example, a memory-constrained tester cannot store all previous samples in order to compare them with new ones. In the simultaneous model, where each processor holds its own set of samples, a lot of communication might be needed to compare samples that are held by different processors. In distributed models, such as CONGEST and LOCAL (see [16]), the problem is defined where each player in a network holds a single sample from a distribution (replacing one sample by a constant amount produces similar behaviour). In these models, it is much cheaper for player to compare their samples with those of a neighboring player, than it is to make such a comparison with players that are far away (on the network topology).

To this end, we introduce the notion of *comparison graphs*. A comparison graph is linked to a collision-based tester (or algorithm) as follows: the vertices of the graph are the samples given as input, and the edges are pairs of samples that are being compared. As stated above: in the classic model this graph is typically the complete clique (all pairs of samples are compared). Under constrained models, however, very specific edges (comparisons) are allowed, whereas others are not. For example, if the sample  $s_1$  is given to one player, and the sample  $s_2$  is given to another player in the simultaneous model, no algorithm can presume to compare the two samples.

---

<sup>1</sup> These testers do not count collisions per se, they use additional crucial steps.

Equipped with the notion of comparison graphs, one can define a *collision-based* tester as a couple  $(G, \tau)$ , where  $G = (V, E)$  is the comparison graph that defines which comparisons are being made, and  $\tau$  is a threshold parameter. The algorithm is defined as follows: it counts the amount of collisions observed,  $Z$ , and compares it to a threshold  $T$  that depends on  $\tau$  and the amount of comparisons made (which is  $|E|$ ).

**Reliable collisions-based testers** In this work we introduce a structural theorem concerning with which sets of comparisons are able to inspire a reliable test for uniformity based solely on counting collisions. This is done by observing the comparison graph  $G$ . We formulate sufficient conditions in terms of the graph  $G$ , that guarantee it induces a good tester (when paired with the right threshold parameter  $\tau$ ). It turns out that two properties of a comparison graph  $G$  are key: the first is the number of edges, which represents the amount of comparisons being made; the second one, somewhat surprisingly, is the number of 2-paths in the graph  $G$ , which encapsulates the amount of dependencies between different comparisons being made.

Few of our testers rely on the same type of graph, that pops up multiple times, for different reasons. To this end, we formulate Lemma 4, that specifies the required parameters for a comparison graph of this type to induce a good tester.

## 1.1 Examples of comparison graphs

It is interesting that the number of samples (the measure we usually wish to minimize) does *not* appear as a condition on our comparison graph directly. However, it does play a role indirectly, as simple inequalities connect the three graph quantities (see Section 4.1).

Our structural theorem basically shows that any comparison graph with enough edges, but not-too-many 2-paths induces a good uniformity tester. To better understand the meaning of this, we fix the amount of edges,  $|E|$ , and review a short list of examples for potential comparison graphs. We are interested in the interplay between the amount of vertices (samples), edges (comparisons made) and 2-paths (dependencies created).

**The clique graph.** The standard tester is actually the full clique, comparing each possible pair of samples. In this dense graph we only need  $\sqrt{|E|}$  vertices in order to have  $|E|$  edges. However, many 2-paths (and dependencies) are created along the way as well,  $\Theta(|E|^{3/2})$ . For this specific case, it is already known the two affects can be balanced to obtain optimal (asymptotic) sample complexity.

**Disjoint cliques.** Another interesting graph (used in some sense in [16]) is actually a union of disjoint cliques. This graph turns out to be quite useful. For once, it makes perfect sense in a simultaneous model, where each player process her own samples, sending a short summary to the referee. Surprisingly, it arises in other models as well.

**A perfect matching.** This graph relates to taking a fresh pair of samples each time we wish to make a new comparison, which leaves us with a set of completely independent collision indicators. Indeed, in this graph there are no 2-paths at all. Not only this tester minimizes the dependencies – it actually overdoes it. Doing so, it pays a price in sample complexity: the number of vertices we have is  $2|E|$ , much larger than the clique, for instance.

**The star graph.** With a fixed number of edges, this graph is actually the way to *maximize* the amount of 2-paths and dependencies – which makes it a very poor comparison graph. Indeed, the tester it induces is equivalent to drawing one element from  $P$ , and comparing it to many other samples, assessing the probability of this element. This test is indeed ill-advised against distributions where  $3/4$  of the elements have mass  $1/n$ .

**The full bipartite graph.** Another graph that could be considered is the full bipartite graph,  $G = (V_1 \sqcup V_2, V_1 \times V_2)$ . Here again we have a free parameter (the size  $|V_1|$ , which determines  $|V_2| = |E| / |V_1|$ ). It ranges from a star-graph (for  $|V_1| = 1$ ) to a balanced graph (for  $|V_1| = |V_2| = \sqrt{E}$ ), where the last one functions asymptotically similarly to the clique. It appears that Theorem 3 only gets optimal testers (minimizing the number of comparisons) from these graphs when they are balanced. To some extent, the test of [12] in the streaming model is based on such a graph. However, they use additional steps that seem crucial for the analysis. more details are given in 3.2, where it is also shown that one can test uniformity in the streaming model solely via collisions counting, using a whole different comparison graph.

## 1.2 Models and Results

Our main result is a method that produces well-performing uniformity testers in various models. In this paper we show a list of uniformity testers in different models, specified below. We also show limitations of our method for most of these models, which point towards a conclusion that no better comparison graphs could have been chosen (up to constant factors in the sample complexity of the induced tester). These limitations rely on a conjecture that no matter the shape of the graph, enough comparisons always must be made.

We emphasize that for any model in which a lower bound is known (for any method, not necessarily collision-based testing), the testers produced by our method are optimal, up to  $\text{poly}(1/\epsilon)$  factors. As far as we know, no testers in the literature are tight with the current lower bounds for these specific cases. Thus, it could be the case that collision-based testing achieves optimal results (even in terms of  $\epsilon$ ) for all the models we consider. A more thorough discussion is given in Section 4.2.

Equipped with the structural theorem, we consider various models and devise a uniformity tester in each one. The key idea here is to translate the constraints of each model into the comparisons we are able to perform, or differently put: a structural limitation on the comparison graph. Doing so will guide us how to choose a “good” comparison graph for this specific model. Having a structure in mind, two formalities are left: (i) Prove that calculation of  $Z, T$  (the number of collisions, and the threshold value) can be done in the model; (ii) Calculate our desired complexity measure (which changes from model to model), and optimize the parameters of the chosen graph (e.g., if the graph is a clique, determine the size of the clique).

**Standard processors.** In Section 3.1, we deal with the classic and the simultaneous model. First we use the classic model as a warm-up. Since this is done in [14], we add to the mix a small insight: our framework (which allows adjusting the threshold) actually provides slightly better constants when placing the threshold much lower than  $1/2$  (at roughly  $\tau = 1/9$ ).

For this model, it was shown by [13] that testing with high precision can be done faster than it would have using standard amplification. However, they use estimation of the empirical distance from uniform over the sample set. It is unclear whether collision-based testing is fit for this task. We do not pursue this direction here.<sup>2</sup>

<sup>2</sup> One reason is that our work focuses on the regime that uses new and different comparison graphs, other than the clique graph. This direction would probably involve analysis that is specific for the clique graph, where all samples can be compared with one another.

We then move to the simultaneous case where multiple players each send a short message to the referee, based on their own samples. The referee then needs to output a decision about the underlying distribution. In this model we want to find a good exchange for the number of players, the number of samples each player gets, and the length of the messages. For example, if the messages can be arbitrarily long, each player can send her entire sample set and the problem becomes trivial. For this reason, the two papers to first consider (independently) testing in this model, had a very different focus. In a preliminary version of [5, 6], only the case of a *single sample* per player was considered, and their algorithms indeed rely on different and interesting strategies, but not collision counting – as this strategy is irrelevant for this regime. In [16] a different approach was taken, where all messages were fixed to a *single bit*, but each player gets multiple samples (and in fact, their algorithm does rely on collisions in some sense, but it does not count them accurately, and does not fall under the umbrella of our definition for collision-based testers).

For our use, as oppose to both these view, we allow both parameters to be larger. We allow multiple samples per player, and show that using a short message (not a single bit, but not much longer), one can devise an efficient tester.

We also consider the asymmetric cost variant of this model, which aims to model the case where not all players have the same sampling capabilities (say, one player might gather samples much faster than another). As we measure our complexity by number of samples, we might as well think of the sampling process as being the bottleneck which we wish to hasten. When trying to minimize the sampling *time* instead, it is only natural to generalize the model to the case where some players are more efficient than others. Other than this motivation, a more technical motivations exists: a natural reduction from the LOCAL model. The LOCAL model is a standard interactive model. Uniformity testing in this model was first considered in [16], which gave a very natural reduction to a simultaneous model where players have different sampling rates.

We remark that many works in the simultaneous model consider communication trade-offs, when assigning only a *single sample* for each party. Our method does not currently extend to this framework, although one might consider integrating it with other methods. For example, one method (e.g., in [6]) uses random hash to a smaller domain. One might consider collisions on this domain instead of the original one. these meta-collisions can be counted within the communication constraints.

Some works in this regime (single sample) focus on privacy aspects of testing (e.g., [2, 7]). This line of research should be irrelevant for our method (and even the extension mentioned above), as any detection of a collision (even on a smaller domain) would immediately give away non-trivial information about the sample.

**Memory-constrained processors.** In Section 3.2, we deal with a different type of processors: *memory-constrained* processors. When observing a single processor of this type, we end up with the streaming model (as described in [12]). At least one scenario which motivates the simultaneous model is seemingly very coherent with such constrained processors. Thinking of a network of sensors, or remote devices, gathering samples – it is quite comprehensible that these processors are not only limited by their ability to communicate with the data center, but also by their ability to store the entire data observed between consecutive reports (in this scenario, the data center is the referee performing the test periodically to detect anomalies). For this reason, we also consider the case of a simultaneous model, where each processor has a small memory budget. In this new model, we easily devise again an efficient uniformity tester.



**Testing in an interactive model.** Lastly, in the full version of this paper, we use the structural theorem to show how on certain graphs one can solve uniformity in the CONGEST model faster than what was previously known to be possible. Specifically, if the communication network has  $k$  players and diameter  $D$ , and each players start with one sample, the best known algorithm runs in  $O(D + n/(k\epsilon^4))$  rounds [16]. The improvement we suggest is an algorithm that takes  $O(D)$  rounds, and works in specific networks that have a certain topological characteristics. Moreover, we show a simple detection procedure of  $O(D)$  rounds, that can be used to recognize such a good topology. This means that *any* network can use the detection procedure first, and then either proceed as in [16], or switch to the faster ( $O(D)$ ) algorithm whenever it is guaranteed to perform well.

### 1.3 Related Work

The task of uniformity testing, as well as the collision-based tester for it, were introduced in [9] (and implicitly in [21]). Later on, upper and lower bounds on the sample complexity of the problem were given by [24] (for most values of  $\epsilon$ ), showing that the optimal sample complexity is in fact  $s = \Theta(\sqrt{n}/\epsilon^2)$ , where  $n$  is the world size, and  $\epsilon$  is a proximity parameter. A preliminary version of [26] giving a tester that achieves this complexity for any value of  $\epsilon$ . The last two papers used two different testers: the first relied on the number of *distinct* elements in the sample set (this test is somewhat dual to counting collisions), and the second on a modified  $\chi^2$  tester. It was then shown by [14] that the collision-based tester does in fact achieve optimal sample complexity too.

In the past few years there has been a growing interest in distribution testing under various computational models, including collaborative testing in multiparty model, the streaming model, privacy aspects of testing, and others ([5, 6, 16, 8, 2, 22, 3, 4, 1, 23, 17, 7] and more). We mention in more details the works concerning uniformity testers in models we pursue.

In a preliminary version of [5, 6] and [16] each, independently, the task of uniformity testing was considered in a simultaneous communication model. In this model, all players receive samples from the same global distribution  $P$ , and all players report to a referee based on their own samples. The referee in turn uses the reports to output (with high probability) whether  $P$  holds some property.

In both lines of work, the focus was uniformity testing, but using two different perspectives: the former zeroes in on one sample per player, where the trade-off in question is between the number of bits each player is allowed in his report, and the number of players needed to the testing process. We think of the number of bits as too small to describe the sampled element fully. In this setting, a full description of two samples is never available to a single player (not even the referee). We do note that the tester given there relies on a looser notion of collisions (Taking a coarser division of  $[n]$  into subsets).

In the later, the focus is different: one now fixes instead the communication to *one* bit per player. Now, the trade-off is between the number of players and the number of samples each one of them takes. The upper bound devised there discuss each player taking the right amount of samples, and notifying the referee 0 if no collision occurred, and 1 otherwise. In some sense, the referee ends up counting collisions (notice the count is trimmed, as one player might see more than 1 collision, but is only able to report 0 or 1). This tester is then used as a black-box to solve uniformity testing in the classic distributed models (CONGEST and LOCAL), in a setting where each player initially draws one sample.

One other paper to specifically discuss uniformity testing is [12], in which a streaming version of the problem is defined, as well as another distributed version, in a blackboard model, where all players are privy to the messages sent by others. As opposed to the

simultaneous models mentioned above – here several rounds of communication are allowed. As it turns out, the streaming algorithm, as well as another algorithm for the distributed version, boil down yet again to counting collisions under the limitations of the model.

In addition to these, quite a few works had the focus of showing impossibility results both in the classic and the entire variety of models. For our interest, we mention [15, 24] for the classic version, as well as [22] for the simultaneous model (with multiple samples per machine), and [12, 3] for the streaming model.

## 2 Preliminaries

Throughout this text, we discuss uniformity testing using collision-based testers.

We let  $[n] := \{1, 2, \dots, n\}$ , and use  $\Delta([n])$  to denote the set of distributions over the set  $[n]$ . As we only care about the support size (rather than the values), it is enough to consider  $P \in \Delta([n])$  as possible input distributions. For a distribution  $P$  over  $[n]$ , we write  $P_i$  for the probability of the  $i^{\text{th}}$  element.

An  $(n, \epsilon)$ -uniformity tester is an algorithm  $\mathcal{A}$  that given oracle access to some unknown distribution  $P \in \Delta([n])$ , takes  $q = q(n, \epsilon)$  samples from  $P$  and satisfy the following:

- If the input distribution is  $P = U_n$ , the uniform distribution over  $[n]$ , then  $\mathcal{A}$  outputs YES with probability at least  $3/4$ .
- If  $\|P - U_n\| \geq \epsilon$ , which means the distribution  $P$  is  $\epsilon$ -far from uniform, then  $\mathcal{A}$  outputs NO with probability at least  $3/4$ .

The distance used here is  $L_1$ . Meaning, for two distribution  $P, Q$ , the distance is  $\|P - Q\| = \sum_{i=1}^n |P_i - Q_i|$ .

To formalize our notion of a *collision-based* tester, we take a fresh point of view of the sampling process. The key object in our analysis is the *comparison graph*, which is simply an undirected graph  $G = (V, E)$ , where we think about  $V$  as a set of placeholders for samples and  $E$  as the pairs of samples which are chosen to be compared with one another.

► **Definition 1** (Sampling process, collision indicators). *Given a comparison graph  $G = (V, E)$  and an input distribution  $P \in \Delta([n])$ , the sampling procedure  $S_P$  is described as a random labeling of the vertices according to  $P$ . We denote by  $S_P : V \rightarrow [n]$ , the process for which  $\forall i. S_P(v_i) \sim P$ , independently from one another. We end up with  $S_P(v_1), \dots, S_P(v_{|V|})$  which is a set of  $|V|$  i.i.d samples from  $P$ .*

Moreover, we define for any edge  $e = (u, v)$  in  $E$ , the collision indicator  $\mathbb{1}_e^P := \mathbb{1}_{S_P(u)=S_P(v)}$ .

When  $P$  is clear from context, we simply write  $S(u)$  for the sample sitting in vertex  $u$ , and  $\mathbb{1}_e$  for the collision indicator.

We are now ready to give a formal definition for a collision-based tester, which relies on a set of comparisons (not necessarily between all pairs of samples), and compares  $Z$  - the amount of collisions, with some threshold value  $T$ .

► **Definition 2** (Collision-based tester). *Fix  $n, \epsilon$ . For any comparison graph  $G = (V, E)$  and real number  $0 \leq \tau \leq 1$ , we define the algorithm  $\mathcal{A} = (G, \tau)$  as follows: upon receiving as input  $|V|$  i.i.d samples from  $P$  (given by  $S_P(v_1), \dots, S_P(v_{|V|})$ ), it computes the following:*

$$Z := \sum_{e \in E} \mathbb{1}_e, \quad T := |E| \cdot \left( \frac{1 + \tau \epsilon^2}{n} \right),$$

and outputs YES if  $Z < T$ , and NO otherwise.

The restriction  $\tau \in [0, 1]$  will help us deal with technicalities, but we note that it is rather intuitive. Indeed, as we will see later, only for these values the expectation of  $Z$  is lower than  $T$  for the good input (uniform distribution), and higher than  $T$  for *all* bad inputs.

Throughout, we will focus on properties of the graph and of our input distribution. We denote by  $|E_G|$ ,  $|V_G|$  the number of edges and vertices in  $G$ , and by  $c(G)$  the number of times a 2-path appears as a subgraph in  $G$ . We count each 2-path twice, for its 2 automorphisms, and so we need to count “directed” 2-paths (so  $e_1, e_2$  and  $e_2, e_1$  are both counted). Formally, we can write  $c(G) = \left| \left\{ (u, v, w) \in \binom{V}{3} \mid (u, v), (v, w) \in E \right\} \right|$ .

For the distribution  $P$  over  $[n]$ , with  $P_i$  for the probability of the  $i^{\text{th}}$  element, we denote the collision probability  $\mu_P = \sum_{i=1}^n P_i^2$ , and the three-way collision probability  $\gamma_P = \sum_{i=1}^n P_i^3$ . For brevity, whenever  $G$  or  $P$  are clear from context, we simply write  $|V|$ ,  $|E|$ ,  $\mu$ ,  $\gamma$ .

## 2.1 Models of Computation

In all our results we are concerned with distribution testing (and specifically uniformity testing), and we deal with various models. Therefore, in the following lines we specify in which way samples are taken in each model, and in what way the answer of the algorithm needs to be declared (where a good tester is the one that outputs YES (resp. NO) with high probability whenever the samples are taken from a YES (resp. NO) distribution).

**The centralized model.** The centralized model is the classic model. In this model one processor receives all samples  $s_1, \dots, s_q$  (in comparison graph notations, we think of  $s_i = S(v_i)$ , and  $q = |V|$ ), and is tasked with outputting a proper answer according to the underlying input distribution. The complexity measure we wish to minimize in this model is  $q$ , the number of samples.

**The simultaneous model.** The second model we consider is the simultaneous model, where  $k$  players (processors) each draw individual samples unseen by all other players. Each player sends a short message to the referee. The referee then aggregates the messages and outputs the answer. Formally, each player is tasked with  $V_i$  where the whole set of vertices in  $G$  is  $V = \sqcup_{i=1}^k V_i$ . The samples of processor  $i$  are then  $\{S(v)\}_{v \in V_i}$ . Each processor can send a message  $a_i$  which is a function of its samples, and a referee receives all messages  $a_1, \dots, a_k$  and outputs the answer. The simultaneous first appeared in the context of testing independently in [16] and preliminary version of [5, 6], where in the first  $|a_i| = 1$  meaning each player is allowed to send one bit, and in the latter  $|V_i| = 1$  meaning each player gets exactly one sample. We take the same point of view as in [16], but we remove the restriction of 1 bit and allow a longer (but still short) message instead.

The number of players  $k$  is given as a parameter, and our goal is to minimize the number of samples *per player*, where all players get the same amount of samples:  $q/k$  (we think of it as sort of parallelization of the sampling process). We also consider the asymmetric-cost variant, where each player has an individual cost for each sample it draws (we think of this cost as the time it takes to draw each sample), and we wish to minimize the cost (or time) of the entire sampling process.

We also discuss the case of memory-constrained processors, also referred to as the streaming model, where this distribution testing was recently considered in [12].

**Memory-constrained processor.** In the memory-constrained model, each processor receives its samples as a stream, and once a sample is dealt with it is gone forever (this is the one-pass variant of the streaming model). A processor can only use a limited amount at each given moment, denote by  $m$  (and measured by memory bits). We think of  $m' = \lfloor m/(2 \log n) \rfloor$  as the number of samples we can store with half the memory (we leave the other half for other calculations). The complexity measure of this model is the number of samples needed to complete the testing task.

We also consider a simultaneous model where each processor is memory-constrained, receiving its samples as a stream, and using its  $m$  bits of memory it needs to come up with a message  $a_i$  to send to the referee once all samples are seen. The referee then receives all messages and outputs an answer. We stick to the case where all processors are of the same type and therefore have the same constraints of  $m$  bits.

### 3 Results

In this section we go over numerous applications of our method. For each model we go over the same phases: we start with intuition as to which comparison graph  $G$  is fit to this model, and we go on to show how one can simulate a collision-based algorithm  $\mathcal{A} = (G, \tau)$ . By simulating the algorithm we mean that by the end of the calculation, some processor will be able to compute both the number of collisions ( $Z$ ) and the fitting threshold ( $T$ ), so it is able to output the answer. The last part is optimizing parameters, where first order parameters are those of  $G$ , and in some application we also give focus to second order parameters (choosing  $\tau$ ).

The strength of the method comes from the following structural theorem that gives sufficient conditions for a comparison graph inducing a good uniformity tester:

► **Theorem 3.** *Fix a domain size  $n$  and a proximity parameter  $\epsilon$ . If the following hold for an algorithm  $\mathcal{A} := (G, \tau)$ :*

1.  $|E| \geq \frac{4n}{\tau^2 \epsilon^4}$ ,
2.  $|E| \geq \frac{16n}{(1-\tau)^2 \epsilon^4}$ , and
3.  $\frac{c(G)}{|E|^2} \leq \frac{(1-\tau)^2 \epsilon^2}{16\sqrt{n}}$ ,

*then  $\mathcal{A}$  is an  $\epsilon$ -uniformity tester.*

We direct the reader to the full version for the proof. For the most part, it is a generalization of the one used in [14] to show the original collision-based tester is optimal (in our notations, the original tester over  $q$  samples is simply the algorithm  $\mathcal{A} = (K_q, 1/2)$ ). While generalizing the proof we leave not one, but *three* separate conditions on a general comparison graph, that together guarantee it induces an  $(n, \epsilon)$ -uniformity tester. This supplies a better, multi-dimensional understanding of how well a collision-based algorithm is guaranteed to perform. We note that if one is willing to ignore constants, one could simply fix  $\tau = 1/2$  and merge the first two conditions into one. However, interestingly for our method other values of  $\tau$  (usually smaller) guarantee slightly better constants. We leave all 3 conditions separate to maintain maximum flexibility when proving application of the theorem.

A specific comparison graph that is key to our algorithms is the one of *disjoint cliques*. Indeed, our strategy will be “perform any comparison you can” which sometimes simply means we have several bulks of samples, where in each bulk all pairs can be compared. To this end, we also give a more specific version of Theorem 3, for graphs that have this structure:

► **Lemma 4.** Fix  $n, \epsilon$ . Fix a comparison graph  $G = \bigsqcup_{i=1}^{\ell} G_i$ , where each  $G_i$  is isomorphic to  $K_q$ , for some  $q \geq 3$ . If the following hold for an algorithm  $\mathcal{A} := (G, \tau)$ :

1.  $q\sqrt{\ell} \geq \frac{\sqrt{12}\sqrt{n}}{\tau\epsilon^2}$
2.  $q\sqrt{\ell} \geq \frac{\sqrt{48}\sqrt{n}}{(1-\tau)\epsilon^2}$
3.  $q\ell \geq \frac{24\sqrt{n}}{(1-\tau)^2\epsilon^2}$

then  $\mathcal{A}$  is an  $\epsilon$ -uniformity tester that uses  $|V|$  samples.

Here again we leave the 3-conditions version in order to be able to adjust the threshold parameter for optimizations. However, here all 3 conditions are quite similar, pointing to the following simple corollary:

► **Corollary 5.** Fix  $n, \epsilon$ . Fix a comparison graph  $G = \bigsqcup_{i=1}^{\ell} G_i$ , where each  $G_i$  is isomorphic to  $K_q$ , for some  $q \geq 3$ . For each constant  $\tau$ , the algorithm  $\mathcal{A} := (G, \tau)$  is an  $\epsilon$ -uniformity tester, if it holds that

$$q\sqrt{\ell} \geq 35\sqrt{n}/\epsilon^2$$

The proofs of these are also omitted, and appear in the full version of the paper. We go on to show how these statements are used to devise testers in various models.

## 3.1 Standard Processors

### 3.1.1 Centralized Model

As a warm up, we re-prove the original collision-based tester works, in term of the comparison graph, and using 4. To add a small twist, we show that under our analysis, better sample complexity is guaranteed when using a biased threshold (meaning, taking  $\tau \neq 1/2$ ). Let us denote  $q := |V|$ , which is the number of samples drawn, and our complexity measure for the model. The following is rather straightforward:

► **Corollary 6.** One can test uniformity using  $q = \Theta\left(\frac{\sqrt{n}}{\epsilon^2}\right)$ .

**Proof.** We simply use the lemma for  $\ell = 1$  (one big clique), and we get 3 conditions of similar form, and in particular:

$$q \geq \max \left\{ \frac{\sqrt{12}}{\tau}, \frac{\sqrt{48}}{1-\tau}, \frac{24}{(1-\tau)^2} \right\} \cdot \frac{\sqrt{n}}{\epsilon^2}$$

Which means that choosing e.g.,  $q = \frac{100\sqrt{n}}{\epsilon^2}$  with  $\tau = 1/2$ . We note that the tester can easily compute  $Z, T$  and therefore execute the collision-based algorithm  $(G, \tau)$ , for any value of  $\tau$ .

An added perk here, is that one can optimize  $\tau$  over the three conditions to reduce sample complexity by a constant factor. Even though the guaranteed constant is somewhat of an artifact of the proof, it is still interesting to see that  $\tau = 1/9$  would reduce the constant from 100 to roughly 35 (while simple optimization over  $\tau$  would reduce it even a bit more, for some irrational threshold value). ◀

### 3.1.2 Simultaneous Model

In the simultaneous model,  $k$  players are each given oracle access to the distribution  $P$ . After taking samples, each player is allowed to send a short message to a referee, who then needs to output the right classification for  $P$  (with high probability).

We give our focus to the variant posed in [16]: what is the number of *samples per player* needed to test uniformity? The “single collision” algorithm devised for that question only requires *one bit* from each player, and indeed it was shown to be optimal in [22]. However, this algorithm is somewhat delicate: first, the range of the parameter  $k$  is limited (it cannot be too high or too low, with regards to  $n, \epsilon$ ); second, if the number of players  $k$  is not accurately known to all players, the algorithm breaks. This is because unlike most results in the classic setting, the analysis here actually requires each player to use a specific amount of samples, but not more than that. Because the players can only communicate a single bit, everything else must be set in advance given the problem’s parameters.

To that end, we relax the model, and allow each player to send more than a single bit, but still only a small number of bits is allowed. These would allow each player to send the number of collisions she saw (rather than *whether* a collision occurred). Our algorithm works for any parameter  $k$ , and can adjust to the case where each player is not exposed to the exact value of  $k$ , but rather to an approximation of it.

The model at hand imposes very concrete limitations on our comparison graph: one cannot compare a sample from one process to a sample of the another process. This means having  $k$  players solving the problem in a parallel way, is equivalent to having a comparison graph whose number of connected components is *at least*  $k$ . Followed by the intuition of “compare every pair you have”, the graph we use is the disjoint cliques graph, with  $k$  disjoint cliques. Our measure of complexity is the number of samples each player used,  $q'$ , which is translated to be the size of each one of our cliques.

► **Corollary 7.** *In the simultaneous model, one can test uniformity using  $q' = \Theta\left(\frac{\sqrt{n}}{\sqrt{k}\epsilon^2}\right)$  samples per player, where each player is allowed to send  $\Theta(\log(1/\epsilon))$  bits to the referee.*

**Proof.** We use Lemma 4, where each player has an independent sample set of size  $q'$  and compares all the pairs. We have  $k$  players in total doing so.

Our first goal is to show how to simulate a collision-based tester  $(G, \tau)$  in this model. This will be possible whenever  $G$  is made of at least  $k$  vertex-disjoint connected components  $G = \bigsqcup_{i \in [k]} G_i$ .

Now, we can write  $Z = \sum_i Z_i$  where  $Z_i := \sum_{e \in E_i} \mathbb{1}_e$ , and each  $Z_i$  is calculated by the  $i$ th player and then sent to the referee who simply sums them up to produce  $Z$ . Computing  $T$  is easy: the algorithm is known to all, and specifically the values  $\tau, |E|$  are known to the referee. Now she simply needs to output the decision  $\mathbb{1}_{Z < T}$ .

Our next step is to quantify the communication and sampling cost of said algorithm. We aim at a communication cost of roughly  $\log(1/\epsilon)$  bits per player. We note that  $Z_i$  can theoretically be very large, however if it surpasses  $T$ , then obviously  $Z > T$ , and the referee can reject. To this end we designate a specific string to say “too large” and instead each player sends  $Z_i$  only up to a value of  $T$ . This amount can be communicated using merely  $\log(T)$  bits by each player.

We now calculate the sample complexity, measured in samples *per player*. We use Lemma 4 with  $q'$  and  $\ell = k$ , to know that it is enough if we satisfy:

1.  $q' \geq \frac{\sqrt{12}\sqrt{n}}{\tau \cdot \sqrt{k}\epsilon^2}$
2.  $q' \geq \frac{\sqrt{48}\sqrt{n}}{(1-\tau) \cdot \sqrt{k}\epsilon^2}$
3.  $q' \geq \frac{24\sqrt{n}}{(1-\tau)^2 k \epsilon^2}$

Which is enough to show the asymptotic sample complexity we desire.<sup>3</sup>

<sup>3</sup> Assuming large enough  $k$ , the third condition pose asymptotically weaker requirement, and so optimizing  $\tau$  on the first two would yield the optimal guarantee for  $\tau = 1/3$ , getting us to  $q' = \left\lceil \frac{\sqrt{108}\sqrt{n}}{\sqrt{k}\epsilon^2} \right\rceil$  samples per player.

Since the third condition is looser than the first two, we actually choose the number of samples per player,  $q'$ , such that  $|E| = \Theta(n/\epsilon^4)$ . Since  $\tau\epsilon^2 \leq 1$ , this means that our threshold is not too large

$$T = \Theta\left(\frac{n}{\epsilon^4}\right) \cdot \left(\frac{1 + \tau\epsilon^2}{n}\right) = \Theta(1/\epsilon^4),$$

and so the simulation of the tester only requires each player to send  $\log(\Theta(1/\epsilon^4))$  bits to describe the number of collisions she saw.  $\blacktriangleleft$

► **Remark 8.** As apparent from [5, 6, 22], when messages are  $r$  bits long, the correct value to look for in the sample complexity is  $2^r$  (and sometimes  $\sqrt{2^r}$ ). For this reason, when discussing communication, we explicitly write expressions such as  $\log(\Theta(1/\epsilon^4))$  instead of a more general  $\Theta(\log(1/\epsilon))$ . This would prove useful when comparing our results to known lower bounds, as is done in Section 4.2.

### 3.1.3 Asymmetric-Cost Model

This variant is best described and motivated as follows: we think of the sampling process as time consuming, and each of the  $k$  players now has her own sampling rate, meaning that some players are able to draw samples faster than others. We describe the sampling rate vector  $R = (R_1, \dots, R_k)$ , and we let each player draw her own number of samples  $(s_1, \dots, s_k)$ . The complexity measure is the time dedicated for the sampling process, denoted  $t$ . Within  $t$  time, player  $i$  collects exactly  $q_i := \lfloor R_i \cdot t \rfloor$  samples.

Simulating the tester (calculations of  $Z, T$ ) works just as before. The big difference is that now each player has a different amount of samples (depending on her rate  $R_i$ ) and thus a different clique size. We omit the calculations which can be found in the full version, and merely state the result we obtain:

► **Corollary 9.** *In the simultaneous model, one can test uniformity in time  $t = \Theta\left(\frac{\sqrt{n}}{\epsilon^2 \|R\|_2}\right)$ , where  $\|R\|_2 = \sqrt{R_1^2 + \dots + R_k^2}$ , and each player sends  $\Theta(\log(1/\epsilon))$  bits to the referee.*

It is interesting that this tester actually generalizes the previous two. Indeed, taking rate vectors  $R = (1, \dots, 1)$  gives the symmetric model, and  $R = (1, 0, \dots, 0)$  the centralized one.

## 3.2 Memory-Constrained Processors

### 3.2.1 Centralized Model with Memory Constraints

In the streaming version of the problem, introduced in [12], the samples from the distribution  $P$  arrive in a stream, and our processor can only store  $m$  bits of memory. We wish to stream as few samples as possible and still output with high probability whether  $P$  is uniform or  $\epsilon$ -far. For our purposes, we use  $m' = \lfloor m/2(\log(n)) \rfloor$ , the number of *samples* that can be stored with half of the memory space (the other half will be used to perform calculations).

The restriction this model imposes is that when a certain sample is being processed, it can only be compared with the  $O(m')$  samples that are currently stored. This intuition can be formalized, showing that any comparison graph used in the streaming model must have a bounded *average* degree (the maximal degree can be arbitrarily high: store the first sample and compare with all others). Indeed, we later quantify this statement (see Claim 16, in Section 4.1.2). For now, we just use it as intuition for the right comparison graph.



In [12] the full bipartite graph was used (with few modifications): they store a batch of samples  $V_1$  and compare the rest of the stream ( $V_2$ ) while comparing each new samples to all samples in  $V_1$ . For a partial range of  $m$ , they achieve sample complexity of  $\Theta(n/(m'\epsilon^4))$ , which is shown to be optimal for an even more limited range parameter  $m$ . It was left as an open question whether this sample complexity can be attained for a wider range of values for  $m$ . For the upper bound, we answer this question in the positive. We use not only a different analysis, but rather a whole different comparison graph, one that also has a low average degree. For simpler representation, we again turn to the disjoint cliques graph used before. The induced algorithm is this: we allocate half the memory for samples and half for calculations. for the first half we take batches of  $m'$  samples, and make all comparisons in between them. We then delete them entirely and make room for a new batch. The number of comparisons is a compact information, that can be easily calculated and stored on the second half of our memory tape.

Once again we leave the full details in the full version of this paper and sum up the differences. The calculations are somewhat similar to before. The one major difference (that results with a different-looking result) is the fact we now express the sample complexity with respect to parameter  $m$  (the size of each clique), whereas before we used  $k$  (the number of cliques) as a parameter.

Since the first half of the tape must store samples, and the second half must count collisions all the way up to the threshold  $T = \Theta(1/\epsilon^4)$ , the result only applies for  $m = \Omega(\log(1/\epsilon))$ . Whenever  $m' \geq c\sqrt{n}/\epsilon^2$  there is not need (nor a possibility) to split our samples to batches, and instead we use one large batch to test uniformity such as in the centralized (standard) model. Combining all the above, we obtain the following result:

► **Corollary 10.** *In the streaming model with memory  $m$ , one can test uniformity using*

$$q = \Theta\left(\max\left\{\frac{n \cdot \log(n)}{m\epsilon^4}, \sqrt{n}/\epsilon^2\right\}\right)$$

*samples, as long as we have  $m = \Omega(\log(1/\epsilon))$ .*

Note that our result has similar complexity as in [12], but it widens the range of acceptable values for  $m$ . They had the requirement of  $m = \Omega(\log(n)/\epsilon^6)$ , which is far larger than the humble requirement posed on our tester. We leave open an interesting question posed also in [12]: is it possible to test for uniformity in the scarce regime, and if so – what is the sample complexity required to do so? (The scarce regime is essentially  $m = o(\log(n))$ ). Though, to comply with our result - e.g., for absurdly small values of  $\epsilon$  - it can be redefined to  $m = o(\max\{\log(1/\epsilon), \log n\})$

### 3.2.2 Simultaneous Model with Memory Constraints

The main difficulty in this section seems to be handling the multiple parameters: on top of  $n, \epsilon$ , we use  $k$  processors, each of which can store at most  $m$  bits of memory (or  $m'$  samples), and is allowed a short one-sided communication to the referee.

We state the result and omit the proof (which is given in the full version).

► **Corollary 11.** *In the memory-constrained simultaneous model, whenever  $m = \Omega(\log(1/\epsilon))$  bits of memory are allowed for each player, and  $\log(1/\epsilon^4)$  bits of communication are allowed for each player, uniformity can be tested using*

$$\Theta\left(\frac{n}{k\epsilon^4} \cdot \max\left\{\frac{\log(n)}{m}, \frac{\sqrt{k}\epsilon^2}{\sqrt{n}}\right\}\right)$$

*samples per player.*

Note that in the regime of very small memory constraint, the problem suddenly “parallelize” perfectly: imagine taking  $k \leq \Theta((n/\epsilon^4) \cdot (\log^2(n)/m^2))$  machines with memory  $m$  per machine. In this case, each machine needs only a  $1/k$  fraction of the amount of samples required for a single machine with memory  $m$ . This phenomena is explained by the observation that the real barrier (or scarce resource) in this regime is the overall storage, which indeed grows linearly with the number of machines used.

## 4 Limitations of the Method

In this section our goal is to better understand the possibilities (and impossibilities) of collision-based testing, in comparison to arbitrary testing methods (many of which were developed for specific testing task, as mentioned earlier in the text).

Here, we use Definition 2 to shift the discussion to graph terminology, focusing on the comparison graph. We leverage simple graph properties to show some limitations that apply when generating testers, such as done in Section 3.

The results hereinafter apply to testers where correctness is proven via our structural theorem. However, we next formulate a simple conjecture (regarding the necessity of making enough comparisons) that implies these results are true for any tester that answers Definition 2. In matter of fact, we will formally show that under the framework of Theorem 3, the comparison graphs we have chosen are essentially the best to answer the requirements. If the conjecture is proven to be true, we get the stronger results that this graphs are indeed optimal with respect to any tester from Definition 2).

In the discussion at the end of this section, we compare these results with known lower bounds (for arbitrary testers) under the same constraints. This section aims to both show optimality (or near-optimality) of the testers we develop, and more importantly: give a better understanding of collision-based testing and how strong it is, compared to an arbitrary tester.

We first go on to show some basic inequalities that hold in any simple graph (and in our comparison graphs as well). These will serve us for the rest of this section. The goal is to establish the inherent connection between the 3 sizes:  $|V|, |E|, c(G)$  in any simple graph  $G$ .

### 4.1 Conditional Impossibility Results

We start by stating the following easy lemma that connects our quantities of interest in any simple graph.

► **Lemma 12.** *For any simple graph  $G$ , it holds that:*

1.  $|E| \leq \frac{|V|^2}{2}$
2.  $|V| \geq \frac{4|E|^2}{2|E|+c(G)}$
3. *if  $|V| \leq |E|$  then  $c(G) \geq 2|E|$*
4. *Whenever  $|V| \leq |E|$ , we have  $|V| \cdot c(G) \geq 2|E|^2$*

We omit the proof here (it appears in the full version), and proceed to put these properties into use.

In order to understand the scope of possible applications of Theorem 3, we stick to graph notations, and combine the 3 different types of assertions concerning our comparison graph:

1. The requirements needed to apply Theorem 3.
2. The basic graph properties of Lemma 12.
3. Individual assertions that apply for the model at hand.

We note that all the results below only use the first requirement of Theorem 3, which asks for the total number of comparisons to be large enough. We believe this requirement to be inherent for *any* collision-based tester (rather than an artifact of our analysis). We formulate this belief as the following conjecture, which would strengthen the impossibility results of this section to be independent of the analysis (instead of limitation of Theorem 3, we get a lower bound for all collision-based testers as defined in Definition 2).

► **Conjecture 13.** *Any collision-based tester as in Definition 2 that tests uniformity with error at most  $1/4$  must have  $|E| = \Omega\left(\frac{n}{\epsilon^4}\right)$*

Note that it holds in two extremes: for maximum amount of dependencies (clique graph) it follows from any lower bound in the classic version of the task. On the other side, the perfect matching has no dependencies at all, and it induces a tester that draws a new pair of samples each time. As the collision probability is either  $1/n$  or larger than  $(1 + \epsilon^2)/n$ , each comparison is like an independent coin toss with this biased (as observed by [6]). It is known for this problem that at least  $\Theta(n/\epsilon^4)$  tosses are needed to decide the bias, which shows correctness of the conjecture for this comparison graph.

#### 4.1.1 Standard Processors

**Centralized model.** As a warm-up, we turn to the classic model, where no constraints are involved. We easily see that Conjecture 13 helps. Using the first bullet in Lemma 12 gives the known lower bound in the classic version of the problem:  $q = |V| \geq \sqrt{2|E|} = \Omega(\sqrt{n}/\epsilon^2)$ .

**Simultaneous model.** In the simultaneous case, the main restriction is that of each player has her own comparison graph, as no comparisons can be made between two different players. This leads us to the following:

► **Corollary 14.** *Assuming Conjecture 13 holds, the number of samples per player of any collision-based uniformity tester in the simultaneous model is  $q' = \Omega\left(\frac{\sqrt{n}}{\sqrt{k} \cdot \epsilon^2}\right)$ .*

**Proof.** We recall the sample complexity here is the maximum amount of samples one player draws. Formally, we write the comparison graph as the union of  $k$  disjoint parts:  $G = \bigsqcup_{i \in [k]} G_i$ , where  $G_i = (V_i, E_i)$ , and so the complexity measure is simply  $q' = \max_{i \in [k]} |V_i|$ .

We note, however, that  $|E_i| \leq |V_i|^2/2$  for each component  $G_i$ , and therefore:

$$|E| = \sum_{i \in [k]} |E_i| \leq \sum_{i \in [k]} |V_i|^2/2 \leq k \cdot \max_{i \in [k]} |V_i|^2/2 = k \cdot q'^2/2.$$

Plugging in Conjecture 13, we get

$$q' \geq \sqrt{\frac{2|E|}{k}} = \Omega\left(\frac{\sqrt{n}}{\sqrt{k} \cdot \epsilon^2}\right),$$

which ends the proof. ◀

**Asymmetric cost model.** The more elaborate version of the asymmetric-cost model also impose similar limitations, where the difference comes from the generalized definition of the complexity measure.

► **Corollary 15.** *We observe the asymmetric-cost simultaneous model, with sampling rate vector  $(R_1, \dots, R_k)$ . If Conjecture 13 holds, then any collision-based uniformity tester in this model must use sampling time of  $t = \Omega\left(\frac{\sqrt{n}}{\epsilon^2 \|R\|_2}\right)$ .*

**Proof.** We again use  $G = \bigsqcup_{i \in [k]} G_i$ , where  $G_i = (V_i, E_i)$ . However, the complexity measure is the time  $t$  in which player  $i$  with rate  $R_i$  can obtain  $|V_i| = q_i = t \cdot R_i$  samples.

Again, using the trivial edges-vertices inequality over each component  $G_i$ , we get

$$\forall i. |E_i| \leq |V_i|^2 / 2 = t^2 \cdot R_i^2 / 2$$

and summing all together, we get

$$|E| = \sum_{i \in [k]} |E_i| \leq \sum_{i \in [k]} t^2 \cdot R_i^2 / 2 \leq t^2 \|R\|_2^2 / 2.$$

Joined with Conjecture 13, it concludes the proof

$$t \geq \sqrt{\frac{2|E|}{\|R\|_2^2}} = \Omega\left(\frac{\sqrt{n}}{\epsilon^2 \|R\|_2}\right). \quad \blacktriangleleft$$

### 4.1.2 Memory-Constrained Processors

Here we use a slightly more sophisticated argument, to show that the memory constraint can also be translated to graph notation. We emphasize that our desire is to show limitations of our framework, and so we relax the model and assume that comparisons are made on designated memory cells, in which we can only store  $m'$  element names. In order to count collisions *accurately*, we cannot expect to compress this data further. For example,  $m' = o(\sqrt{n})$  and samples are drawn from the uniform distribution  $P = U_n$ , with 99% we need to write in our memory  $m'$  different elements, and this information cannot be compressed.

We go on to show how the memory constraint translates well:

▷ **Claim 16.** Let us assume a constrained machine can only store  $m'$  elements at a time, and it is able to accurately count collisions on a comparison graph  $G = (V, E)$ . Then it must be the case that  $|E| \leq m' \cdot |V|$ .

*Proof.* w.l.o.g let us name the vertices, or samples, by their order in the stream  $V = \{v_1, v_2, \dots, v_s\}$ , and w.l.o.g let us think of the edges in  $E$  as ordered pairs (We only write  $(i, j) \in E$  for pairs where  $i < j$ ).

Now, we note that at time  $t$ , upon processing the sample  $v_t$ , the memory can only store  $m'$  samples from the set  $s(v_1), \dots, s(v_{t-1})$ . This means that the number of edges in  $E$  of the form  $(v_i, v_t)$  is at most  $m'$ . Now, we can count our (ordered) edges using the second item:

$$|E| = \sum_{v_j \in V} |\{(u, v_j) \in E \mid u \in V\}| \leq \sum_{v_t \in V} m' = |V| \cdot m'$$

which completes the proof. ◀

**Centralized model with memory constraints.** We next apply this claim to give similar lower bounds in the following models.

► **Corollary 17.** Assume Conjecture 13 holds. Any collision-based uniformity tester that can only store at most  $m' = \Theta(m/\log(n))$  samples at any given time, must take at least  $q \geq \Omega\left(\frac{n \log(n)}{m \epsilon^4}\right)$  samples.

**Proof.** As our measure complexity is the total amount of samples,  $q = |V|$ , we can combine the claim above with Conjecture 13 to get:

$$q = |V| \geq \frac{|E|}{m'} = \Omega\left(\frac{n \log(n)}{m \epsilon^4}\right),$$

which ends the proof. ◀

**Simultaneous model with memory constraints.** We observe the combined model of simultaneous model with memory-constrained machines. We again restrict ourselves to a specific framework: each player uses her own graph  $G_i = (V_i, E_i)$ , where memory is allocated for sampled elements, and then all players send a short message to the referee. The entire comparison graph the algorithm is based on is  $G = \bigsqcup_i G_i$ .

► **Corollary 18.** *Any collision-based uniformity tester in a simultaneous model of  $k$  machines that can store up to  $m'$  samples each, must use*

$$q' = \Omega \left( \frac{n}{k\epsilon^4} \cdot \max \left\{ \frac{\log(n)}{m}, \frac{\sqrt{k}\epsilon^2}{\sqrt{n}} \right\} \right)$$

*samples per player, assuming Conjecture 13 holds .*

**Proof.** We start by re-writing the desired expression:

$$q' = \Omega \left( \max \left\{ \frac{\sqrt{n}}{\sqrt{k}\epsilon^2}, \frac{n \log(n)}{mk\epsilon^4} \right\} \right)$$

We show the two lower bounds separately. Indeed, the first lower bound can be derived directly from Corollary 14:

$$q' = \Omega \left( \sqrt{n/(k\epsilon^2)} \right)$$

For the second lower bound, we extend Corollary 17 instead. As each  $G_i$  is done by a machine with memory constraints, we apply Claim 16 to player  $i$  and get  $|V_i| \geq |E_i|/m'$ . We recall that our measure complexity is in fact  $q' := \max_i |V_i|$ , and as the maximum is greater than the average, we get:

$$q' \geq \frac{\sum_i |V_i|}{k} \geq \frac{\sum_i |E_i|}{m' \cdot k} = \frac{|E|}{m' \cdot k}$$

And plugging in Conjecture 13 on the entire graph  $G = (V, E)$ , we get

$$q' = \Omega \left( \frac{n}{m' \cdot k\epsilon^4} \right) = \Omega \left( \frac{n \log(n)}{m \cdot k\epsilon^4} \right),$$

concluding the proof. ◀

## 4.2 Discussion

We point out to the fact that in all 4 models (as well as the classic model), the limitation of this method coincide with the upper bounds we obtained in the previous section. This does not come as a surprise, as choosing the “right” comparison graph is easily made once the constraints of the model are understood. The rule of thumb is simply to compare all pairs that can be compared.

It is more interesting, though, to compare the collection of possible testers with the sub-collection of collision-based testers. We next brief through known impossibility results (for an arbitrary tester. It turns out for the most part, collision-based testers compete well with others. For the classical model, as already established in [13], collision-based testing is in fact optimal. For the two simultaneous models (with no memory constraints), the collision-based testers perform optimally in terms of  $n, k$  (or  $n, (R_1, \dots, R_k)$  in the asymmetric case). To the best of our knowledge, the only testers for these models to consider *multiple* samples

per processor are the ones of [16]. For both models, the two papers show similar sample complexity, but there are two non-trivial differences. On one hand, the new testers use  $\log(\Theta(1/\epsilon^4))$  bits per communication, instead of a single one used in [16]. On the other hand, the new testers work for the full range of the parameter  $k$  (the number of players), whereas the previous results excluded extreme values: e.g., in the symmetric case it only works for  $c/\epsilon^4 \leq k \leq c'n\epsilon^4$ .

Lower bounds for both models are shown in [22], even for the case of  $r$ -bit messages. While the tester of [16] is an optimal one-bit protocol, ours is not known to be optimal  $r$ -bit protocol. This is true as the aforementioned lower bound weakens by a factor  $2^r$  for the longer  $r$ -bit messages. In our case, we have  $2^r = \Theta(1/\epsilon^4)$ , which means there is a gap of  $\epsilon^4$  between the general lower bound, and the optimal collision-based tester we obtain. Despite the gap, no better tester is known for this amount of bits and  $q > 1$  samples per player.

In the streaming model our tester attain the same sample complexity as the best known tester (that of [12]), but for a wider range for the parameter  $m$ . A matching lower bound can also be found in [12], but only for a more restricted range of the parameter  $m$ , whereas for the general case they give a weaker lower bound, which leaves an  $\epsilon^2$  gap between the general case and the optimal collision-based tester.

To the best of our knowledge, the simultaneous model with memory constraints was never considered in the context of distribution testing, and therefore there are no prior results. However, we conjecture that similarly to before, the collision-based tester achieves optimal sample complexity in parameters  $n, k, m$ , with a possible  $\text{poly}(\epsilon)$  gap that would pop, as before, due to the use of longer messages.

---

## References

- 1 Jayadev Acharya, Sourbh Bhadane, Piotr Indyk, and Ziteng Sun. Estimating entropy of distributions in constant space. In *Advances in Neural Information Processing Systems*, 2019.
- 2 Jayadev Acharya, Clément Canonne, Cody Freitag, and Himanshu Tyagi. Test without trust: Optimal locally private distribution testing. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2067–2076, 2019.
- 3 Jayadev Acharya, Clément L Canonne, Yuhua Liu, Ziteng Sun, and Himanshu Tyagi. Interactive inference under information constraints. *arXiv preprint*, 2020. [arXiv:2007.10976](#).
- 4 Jayadev Acharya, Clément L Canonne, and Himanshu Tyagi. Distributed signal detection under communication constraints. In *Conference on Learning Theory*. PMLR, 2020.
- 5 Jayadev Acharya, Clément L Canonne, and Himanshu Tyagi. Inference under information constraints i: Lower bounds from chi-square contraction. *IEEE Transactions on Information Theory*, 2020.
- 6 Jayadev Acharya, Clément L Canonne, and Himanshu Tyagi. Inference under information constraints ii: Communication constraints and shared randomness. *IEEE Transactions on Information Theory*, 2020.
- 7 Kareem Amin, Matthew Joseph, and Jieming Mao. Pan-private uniformity testing. In *Conference on Learning Theory*, pages 183–218. PMLR, 2020.
- 8 Alexandr Andoni, Tal Malkin, and Negev Shekel Nosatzki. Two party distribution testing: Communication and security. In *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2019.
- 9 Tugkan Batu, Lance Fortnow, Ronitt Rubinfeld, Warren D. Smith, and Patrick White. Testing that distributions are close. In *41st Annual Symposium on Foundations of Computer Science, FOCS 2000, 12-14 November 2000, Redondo Beach, California, USA*, pages 259–269, 2000.
- 10 Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-testing/correcting with applications to numerical problems. *Journal of computer and system sciences*, 47(3):549–595, 1993.

- 11 Clément L. Canonne. A survey on distribution testing: Your data is big, but is it blue? *Electronic Colloquium on Computational Complexity (ECCC)*, 22:63, 2015.
- 12 Ilias Diakonikolas, Themis Gouleakis, Daniel M Kane, and Sankeerth Rao. Communication and memory efficient testing of discrete distributions. *arXiv preprint*, 2019. [arXiv:1906.04709](#).
- 13 Ilias Diakonikolas, Themis Gouleakis, John Peebles, and Eric Price. Sample-optimal identity testing with high probability. *CoRR*, abs/1708.02728, 2017. [arXiv:1708.02728](#).
- 14 Ilias Diakonikolas, Themis Gouleakis, John Peebles, and Eric Price. Collision-based testers are optimal for uniformity and closeness. *Chicago Journal OF Theoretical Computer Science*, 1:1–21, 2019.
- 15 Ilias Diakonikolas and Daniel M. Kane. A new approach for testing properties of discrete distributions. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, 2016.
- 16 Orr Fischer, Uri Meir, and Rotem Oshman. Distributed uniformity testing. In *Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing*, PODC '18. ACM, 2018.
- 17 Sumegha Garg, Pravesh K. Kothari, and Ran Raz. Time-Space Tradeoffs for Distinguishing Distributions and Applications to Security of Goldreich’s PRG. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2020)*, pages 21:1–21:18, 2020.
- 18 Oded Goldreich. *Introduction to Property Testing*. Cambridge University Press, 2017.
- 19 Oded Goldreich. The uniform distribution is complete with respect to testing identity to a fixed distribution. In *Computational Complexity and Property Testing*. Springer, 2020.
- 20 Oded Goldreich, Shari Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *Journal of the ACM (JACM)*, 45(4):653–750, 1998.
- 21 Oded Goldreich and Dana Ron. On testing expansion in bounded-degree graphs. *Electronic Colloquium on Computational Complexity (ECCC)*, 7(20), 2000.
- 22 Uri Meir, Dor Minzer, and Rotem Oshman. Can distributed uniformity testing be local? In *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing*, pages 228–237. ACM, 2019.
- 23 Varun Narayanan, Manoj Mishra, and Vinod M Prabhakaran. Private two-terminal hypothesis testing. *arXiv preprint*, 2020. [arXiv:2005.05961](#).
- 24 L. Paninski. A coincidence-based test for uniformity given very sparsely sampled discrete data. *IEEE Transactions on Information Theory*, 54(10):4750–4755, 2008.
- 25 Ronitt Rubinfeld and Madhu Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM Journal on Computing*, 25(2):252–271, 1996.
- 26 Gregory Valiant and Paul Valiant. An automatic inequality prover and instance optimal identity testing. *SIAM Journal on Computing*, 46(1):429–455, 2017.



# Circular Trace Reconstruction

Shyam Narayanan

Massachusetts Institute of Technology, Cambridge, MA, USA  
shyamsn@mit.edu

Michael Ren

Massachusetts Institute of Technology, Cambridge, MA, USA  
mren36@mit.edu

---

## Abstract

*Trace reconstruction* is the problem of learning an unknown string  $x$  from independent traces of  $x$ , where traces are generated by independently deleting each bit of  $x$  with some deletion probability  $q$ . In this paper, we initiate the study of *Circular trace reconstruction*, where the unknown string  $x$  is circular and traces are now rotated by a random cyclic shift. Trace reconstruction is related to many computational biology problems studying DNA, which is a primary motivation for this problem as well, as many types of DNA are known to be circular.

Our main results are as follows. First, we prove that we can reconstruct arbitrary circular strings of length  $n$  using  $\exp(\tilde{O}(n^{1/3}))$  traces for any constant deletion probability  $q$ , as long as  $n$  is prime or the product of two primes. For  $n$  of this form, this nearly matches what was the best known bound of  $\exp(O(n^{1/3}))$  for standard trace reconstruction when this paper was initially released. We note, however, that Chase very recently improved the standard trace reconstruction bound to  $\exp(\tilde{O}(n^{1/5}))$ . Next, we prove that we can reconstruct random circular strings with high probability using  $n^{O(1)}$  traces for any constant deletion probability  $q$ . Finally, we prove a lower bound of  $\tilde{\Omega}(n^3)$  traces for arbitrary circular strings, which is greater than the best known lower bound of  $\tilde{\Omega}(n^{3/2})$  in standard trace reconstruction.

**2012 ACM Subject Classification** Mathematics of computing  $\rightarrow$  Probabilistic algorithms

**Keywords and phrases** Trace Reconstruction, Deletion Channel, Cyclotomic Integers

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.18

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/2009.01346>.

**Funding** *Shyam Narayanan*: Supported by the MIT Akamai Fellowship, the NSF Graduate Fellowship, and a Simons Investigator Award.

*Michael Ren*: Supported by NSF-DMS grant 1949884 and NSA grant H98230-20-1-0009.

**Acknowledgements** The first author thanks Professor Piotr Indyk for many helpful discussions and feedback, Mehtaab Sawhney for pointers to some references, and Professor Bjorn Poonen for a helpful discussion on sums of roots of unity. The second author thanks Professor Joe Gallian for running the Duluth REU at which part of this research was conducted, as well as program advisors Amanda Burcroff, Colin Defant, and Yelena Mandelshtam for providing a supportive environment. The authors also thank Amanda Burcroff for helpful edits on the paper's writeup.

## 1 Introduction

The trace reconstruction problem asks one to recover an unknown string  $x$  of length  $n$  from independent noisy samples of the string. In the original setting,  $x$  is a binary string in  $\{0, 1\}^n$ , and a random subsequence  $\tilde{x}$  of  $x$ , called a *trace*, is generated by sending  $x$  through a deletion channel with deletion probability  $q$ , which removes each bit of  $x$  independently with some fixed probability  $q$ . The main question is to determine how many independent traces are needed to recover the original string with high probability. This question has become very well studied over the past two decades [26, 27, 7, 24, 23, 38, 30, 19, 32, 35, 20, 21, 22, 13, 15, 14],



© Shyam Narayanan and Michael Ren;

licensed under Creative Commons License CC-BY

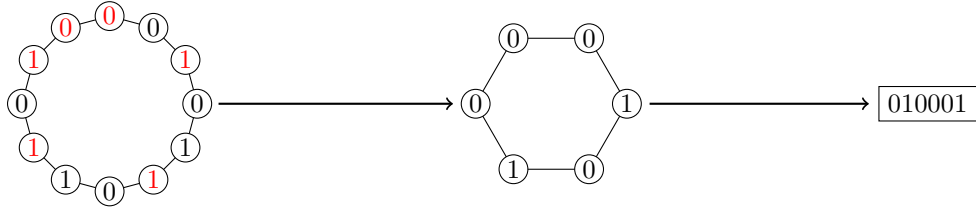
12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 18; pp. 18:1–18:18

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



■ **Figure 1** An example of a circular trace. We start with an unknown circular string (top left). Each bit of the string is randomly deleted (red bits are deleted, black bits are retained) and the order of the retained bits is preserved, so we are left with the smaller circular string. However, since there is no beginning or end of the circular string, we assume the string is seen in clockwise order starting from a randomly chosen bit.

with many results over various settings. For instance, people have studied the case where we wish to reconstruct  $x$  for any arbitrarily chosen  $x \in \{0, 1\}^n$  (worst-case) or the case where we just wish to reconstruct a randomly chosen string  $x$  (average-case). People have also studied the trace reconstruction problem for various values of the deletion probability  $q$ , such as if  $q$  is a fixed constant between 0 and 1 or decays as some function of  $n$ . People have also studied variants where the traces allow for insertions of random bits, rather than just deletions, and variants where the string is no longer binary but from a larger alphabet.

Finally, various generalizations or variants of the trace reconstruction problem have also been developed. These include error-correcting codes over the deletion channel (i.e., “coded” trace reconstruction) [16, 11], reconstructing matrices [25] and trees [18] from traces, and reconstructing mixtures of strings from traces [3, 4, 31].

In this paper, we develop and study a new variant of trace reconstruction that we call *Circular trace reconstruction*. In this variant, there is again an unknown string  $x \in \{0, 1\}^n$  that we can sample traces from, but this time, the string  $x$  is a cyclic string, meaning that there is no beginning or end to the string. Equivalently, one can imagine a linear string that undergoes a random cyclic shift before a trace is returned. See Figure 1 for an example. Our goal, like in the normal trace reconstruction, is to reconstruct the original circular string using as few random traces as possible.

## 1.1 Main Results and Comparison to Linear Trace Reconstruction

Perhaps the first natural question about circular trace reconstruction is the following: how does the sample complexity of circular trace reconstruction compare to the sample complexity of standard (linear) trace reconstruction? Intuitively, one should expect circular trace reconstruction to be at least as difficult as standard trace reconstruction, since given any trace of a linear string, we can randomly rotate it to get a trace of the corresponding circular string. This reasoning, however, is slightly flawed. For instance, perhaps the hardest instance of linear trace reconstruction comes from distinguishing between two strings  $x$  and  $y$  which are different as linear strings but equivalent up to a cyclic shift. In this case, the circular trace reconstruction problem does not even need to distinguish between  $x$  and  $y$ , because they are equivalent! However, by padding the trace with extra bits before randomly rotating, one can show that circular trace reconstruction is at least as hard as linear trace reconstruction in both the worst-case and average-case. Indeed, we have the following proposition. As its proof is quite simple, we defer it to Appendix A in the full version of this paper on arXiv.

► **Proposition 1.** *Suppose that we can solve worst-case (resp., average case) circular trace reconstruction over length  $m$  strings with deletion probability  $q$  using  $T(m, q)$  traces. Then, we can solve worst-case (resp., average case) linear trace reconstruction over length  $n$  strings with deletion probability  $q$  using  $\min_{m \geq 2n} T(m, q)$  traces.*

Given Proposition 1, any upper bounds for circular trace reconstruction imply nearly equivalent upper bounds for the linear trace reconstruction, and any lower bounds for linear trace reconstruction imply nearly equivalent lower bounds for circular trace reconstruction. This raises two natural questions. First, can we match or nearly match the best linear trace reconstruction upper bounds for circular trace reconstruction? Second, can we beat the best linear trace reconstruction lower bounds for circular trace reconstruction?

The first main result we prove is for worst-case circular strings. At the time of the initial release of this paper, the best known upper bound for worst-case linear trace reconstruction with deletion probability  $q$ , where  $q$  is a fixed constant between 0 and 1, is  $\exp(O(n^{1/3}))$ , where the unknown string has length  $n$  [19, 32]. Shortly afterwards, the upper bound was improved to  $\exp(\tilde{O}(n^{1/5}))$  [14]. Our first main result, which we prove in Section 3, provides an upper bound for the circular trace reconstruction problem that nearly matches the results of [19, 32], but only if the length  $n$  has at most 2 prime factors.

► **Theorem 2.** *Let  $x$  be an unknown, arbitrary circular string of length  $n$ , let  $q$  be the deletion probability of each element in the string, and let  $p = 1 - q$  be the retention probability. Then, if  $n$  is either a prime or a product of two (possibly equal) primes, using  $\exp(O(n^{1/3}(\log n)^{2/3}p^{-2/3}))$  random traces, we can determine  $x$  with failure probability at most  $2^{-n}$ .*

The primary reason why our theorem fails for  $n$  having 3 or more prime factors is that we prove the following number theoretic result which is crucial in our algorithm.

► **Theorem 3.** *For any fixed integer  $n \geq 2$ , the following statement is true **if and only if**  $n$  has at most 2 prime factors, counting multiplicity.*

*Define  $\omega := e^{2\pi i/n}$ , and suppose that  $a_0, \dots, a_{n-1}, b_0, \dots, b_{n-1}$  are all integers in  $\{0, 1\}$ . Also, suppose that for all  $0 \leq k \leq n-1$ , there is some integer  $c_k$  such that  $\sum_{i=1}^n a_i \omega^{i \cdot k} = \omega^{c_k} \cdot \sum_{i=1}^n b_i \omega^{i \cdot k}$ . Then, the sequences  $\{a_i\}$  and  $\{b_i\}$  are cyclic shifts of each other.*

In this conference version, we only prove the above theorem in the special case that  $n$  is prime. For the full proof of this theorem, please see the arXiv version of this paper at <https://arxiv.org/abs/2009.01346>.

The next main result we prove is for average-case circular strings: we show that a random circular string can be recovered using a polynomial number of traces. Formally, we prove the following theorem in Section 4.

► **Theorem 4.** *Let  $x$  be an unknown but randomly chosen circular string of length  $n$  and let  $0 < q < 1$  be the deletion probability of each element. Then, there exists a constant  $C_q$  depending only on  $q$  such that we can determine  $x$  with failure probability at most  $n^{-10}$  using  $O(n^{C_q})$  traces.*

The main lemma we need to prove Theorem 4 is actually a result that is true for worst-case strings. Specifically, we show how to recover the multiset of all consecutive substrings of length  $O(\log n)$  using a polynomial number of traces. While this does not guarantee that we can recover an arbitrary circular string, it does allow us to recover what we will call *regular strings*, which we show comprise the majority of circular strings. The following lemma may be of independent interest for studying worst-case strings as well, as it allows one to gain information about all “consecutive chunks” of the unknown string using only a polynomial number of queries.

► **Lemma 5.** *Let  $x = x_1 \cdots x_n$  be an arbitrary circular string of length  $n$  and let  $0 < q < 1$  be the deletion probability of each element. Then, for  $k = 100 \log n$ , we can recover the multiset of all substrings  $\{x_i x_{i+1} \cdots x_{i+k-1}\}_{i=1}^n$ , where indices are modulo  $n$ , using  $O(n^{C_q})$  traces with failure probability  $n^{-10}$ , where  $C_q$  is a constant that only depends on  $q$ .*

The best upper bound for average-case linear trace reconstruction is  $\exp(O((\log n)^{1/3}))$  [22]. Unfortunately, we were not able to adapt their argument to circular strings. One major reason why we are unable to do so is that in the argument of [22] (as well as [35], which provides an  $\exp(O((\log n)^{1/2}))$  sample algorithm), the authors recover the  $(k+1)^{\text{st}}$  bit of the string assuming the first  $k$  bits are known using a small number of traces, and by reusing traces, they inductively recover the full string. However, since we are dealing with circular strings, even recovering the “first” bit does not make much sense. However, we note that even a polynomial-sample algorithm is quite nontrivial. In the linear case, a polynomial-sample algorithm for average-case strings was first proven by [23], and their algorithm only worked as long as the deletion probability  $q$  was at most some small constant, which when optimized is only about 0.07 [35].

Our final main result regards lower bounds for worst-case strings. For linear worst-case strings, the best known lower bound for trace reconstruction is  $\tilde{\Omega}(n^{3/2})$  [13]. For circular trace reconstruction, we show an improved lower bound of  $\tilde{\Omega}(n^3)$ . Moreover, the proof of our lower bound is actually much simpler and cleaner than those of the known lower bounds for standard trace reconstruction [13, 21]. Specifically, we prove the following theorem, done in Section 5:

► **Theorem 6.** *Let  $x$  be the string  $10^n 10^{n+1} 10^{n+k} = 1 \underbrace{0 \dots 0}_{n \text{ times}} 1 \underbrace{0 \dots 0}_{n+1 \text{ times}} 1 \underbrace{0 \dots 0}_{n+k \text{ times}}$ , where  $n \geq 1$  and  $2 \leq k \leq 4$ . Likewise, let  $y$  be the string  $y = 10^n 10^{n+k} 10^{n+1}$ . Then, the strings  $x, y$  are not equivalent up to cyclic rotations, but for any constant deletion probability  $q$ , one requires  $\Omega(n^3 / \log^3 n)$  random traces to distinguish between the original string being  $x$  or  $y$ . Thus, for all integers  $n$ , worst-case circular trace reconstruction requires at least  $\tilde{\Omega}(n^3)$  random traces.*

### 1.1.1 Concurrent Work

We note that a very similar statement to Lemma 5, but for linear strings, was proven in independent concurrent work by Chen et. al. [15, Theorem 2], which provides a polynomial-sample algorithm for a “smoothed” variant of worst-case linear trace reconstruction. Many ideas in our proof of Lemma 5 and their proof appear to overlap, though our proof is substantially shorter. We discuss the relation between our work and [15] further at the end of Section 4.

## 1.2 Motivation and Relation to Other Work

From a theoretical perspective, circular trace reconstruction can bring many novel insights to the theory of reconstruction algorithms, some of which may be useful even in the standard trace reconstruction problem. For instance, the proof of Theorem 2 combines analytic, statistical, and combinatorial approaches as in previous trace reconstruction papers, but now also uses ideas from number theory and results about cyclotomic integers. To the best of our knowledge, this paper is the first paper on trace reconstruction to utilize number theoretic ideas, though there is work on other problems about cyclic strings that uses ideas from number theory. Also, Lemma 5 shows a way to recover all contiguous sequences in the

original string of length  $O(\log n)$  for arbitrary circular strings, which is a new result even in the linear case (concurrent with [15]) and has applications to problems in linear trace reconstruction as well (as done in [15]).

From an applications perspective, trace reconstruction is closely related to the multiple sequence alignment problem in computational biology. In the multiple sequence alignment problem, one is given DNA sequences from several related organisms, and the goal is to align the sequences to determine what mutations each descendant underwent from their common ancestor: the trace reconstruction problem is analogous to actually recovering the common ancestor. One can learn more about trace reconstruction's relation to the multiple sequence alignment problem (as well as to various other problems in computational biology) via the recent survey [8].

The multiple sequence alignment problem is also a key motivation for studying circular trace reconstruction. Many important types of DNA, such as mitochondrial DNA in humans and other eukaryotes, chloroplast DNA, bacterial DNA, and DNA in plasmids, are predominantly circular (see, e.g., [37, pp. 313, 397, 516-517], or [1]). Therefore, understanding circular trace reconstruction could prove useful in reconstructing ancestral sequences for mitochondrial or bacterial DNA. Another problem in computational biology that trace reconstruction may be applicable to is the DNA Data Storage problem, where data is stored in DNA and can be recovered through sequencing, though the stored DNA may mutate over time [17, 34]. Recently, long-term DNA data storage in plasmids has been successfully researched [33], which further motivates the study of circular trace reconstruction.

Besides the linear trace reconstruction problem, circular trace reconstruction is also closely related to the problem of population recovery from the deletion channel [3, 4, 31], where the goal is to recover an unknown mixture of  $\ell$  strings from random traces. Indeed, receiving traces from a circular string is equivalent to receiving traces from a uniform mixture of a linear string along with all of its cyclic shifts, so circular trace reconstruction can be thought of as an instance of population recovery from the deletion channel with mixture size  $\ell = n$ .

Unfortunately, the best known algorithm for population recovery over worst-case strings requires  $\exp(\tilde{O}(n^{1/3}) \cdot \ell^2)$  traces [31], which is not useful if  $\ell = n$ . However, to prove our worst-case upper bound, we will use ideas based on [19, 32, 31] to estimate certain polynomials that depend on the unknown circular string  $x$ . For the average case problem, i.e., if given a mixture over  $\ell$  random strings, population recovery can be done with  $\text{poly}(\ell, \exp((\log n)^{1/3}))$  random traces. While this seemingly implies a  $\text{poly}(n)$ -sample algorithm for average-case circular trace reconstruction, the  $n$  cyclic shifts of the circular string are quite similar to each other and thus do not behave like a collection of  $n$  independent random strings. Indeed, our techniques for average-case circular trace reconstruction are very different from those developed in [4].

While circular strings have not been studied before in the context of trace reconstruction, people have studied circular strings and cyclic shifts in the context of edit distance [28, 2], multi-reference alignment [5, 6, 36], and other pattern matching problems [12]. We note that [2] also applies results from number theory and about cyclotomic polynomials, though their techniques overall are not very similar to ours.

### 1.3 Proof Overview

In this subsection, we highlight some of the ideas used in Theorems 2, 4, and 6.

The proof of Theorem 2 is partially based on ideas from [19, 32, 31]. The authors of [19, 32] consider two strings  $x, y \in \{0, 1\}^n$  and show how to distinguish between random traces of  $x$  and random traces of  $y$ . To do so, they construct an unbiased estimator for the

polynomial  $P(z; x) := \sum_{i=1}^n x_i z^i$  (or  $P(z; y) = \sum_{i=1}^n y_i z^i$ ) solely based on the random trace of either  $x$  or  $y$ , for some  $z \in \mathbb{C}$ . By showing that the unbiased estimator is never “too” large and that  $P(z; x)$  and  $P(z; y)$  differ enough for an appropriate choice of  $z$ , they can estimate this quantity using random traces to distinguish between  $x$  and  $y$ . In our case, applying the same estimator will give us an unbiased estimator for  $P'(z; x) := \mathbb{E}_i[P(z; x^{(i)})]$ , where  $x^{(i)}$  is the  $i$ th cyclic shift of  $x$ . Unfortunately, it turns out that  $P'(z; x) = P'(z; y)$  as polynomials in  $z$  as long as  $x, y$  have the same number of 1’s, even if  $x$  and  $y$  are vastly different as circular strings. Our goal will then be to establish some other polynomial  $Q(z; x)$  such that we can construct a good unbiased estimator, but at the same time  $Q'(z; x) := \mathbb{E}_i[Q(z; x^{(i)})]$  and  $Q'(z; y) := \mathbb{E}_i[Q(z; y^{(i)})]$  are distinct polynomials for any distinct cyclic strings  $x, y$ . We show that the polynomial  $Q(z; x) := z^{kn} P(z; x)^k P(z^{-k}; x)$  will do the job, for some small integer  $k$ . We provide a (significantly more complicated) unbiased estimator of  $Q(z; x)$  using a random trace: the construction is similar to that of [31], which shows how to estimate  $P(z; x)^k$  for some integer  $k$ . To show that  $Q(z; x) \neq Q(z; y)$  as polynomials, we first show that  $Q(z; x)$  has the special property that if  $z$  is a cyclotomic  $n$ th root of unity, this polynomial is invariant under cyclic shifts of  $x$ ! Thus, it just suffices to show that if  $x, y \in \{0, 1\}^n$  are not cyclic shifts of each other, there is some  $n$ th root of unity  $z = e^{2\pi i r/n}$  for some  $0 \leq r \leq n-1$  such that  $P(z; x)^k P(z^{-k}; x) \neq P(z; y)^k P(z^{-k}; y)$ . This will require significant number theoretic computation, and will be true as long as  $n$  is a prime or a product of two primes.

The bulk of the proof of Theorem 4 will be proving Lemma 5, which reconstructs all consecutive substrings of length  $100 \log n$  in the unknown circular string  $x$ . For a random string  $x$ , these substrings will all be sufficiently different, so once we know the substrings, we can reconstruct the full string because there will only be one way to “glue” together the substrings. Therefore, we focus on explaining the ideas for Lemma 5. Our goal will be to determine how many times a string  $s$  appears consecutively in  $x$  for each string  $s$  of length  $100 \log n$ . For an unknown string  $x$  and  $i$  between 0 and  $n - 100 \log n$ , we let  $c_i$  be the number of times  $s$  appears (possibly non-contiguously) in some contiguous block of length  $i + 100 \log n$  in  $x$ . Then, a basic enumerative argument shows that for a random (cyclically shifted) trace  $\tilde{x} = \tilde{x}_1 \tilde{x}_2 \cdots \tilde{x}_m$ , the probability that  $\tilde{x}_1 \cdots \tilde{x}_{100 \log n} = s$  can be written as  $\sum_{i \geq 0} c_i (1 - q)^{100 \log n} q^i$ , and we wish to recover  $c_0$ . The  $(1 - q)^{100 \log n}$  term is a constant that equals  $1/\text{poly}(n)$ , so it is easy to recover an approximation to  $\sum_{i \geq 0} c_i q^i$ . We truncate this polynomial at an appropriate degree (approximately  $C \log n$  for some large  $C$ ) and show that the truncated polynomial  $\sum_{i=0}^{C \log n} c_i x^i$  is very close to the original polynomial, but differs from  $\sum_{i=0}^{C \log n} c'_i x^i$  for some  $x \in [q, (1 - q)/2]$  by a significant amount, if  $c'_0 \neq c_0$ , using ideas based on [10]. We can also simulate a trace with deletion probability  $x > q$  by taking a “trace of the trace.” This will be sufficient in determining  $c_0$ , and therefore, the (multi)-set of all consecutive substrings of length  $100 \log n$ .

The proof of Theorem 6 proceeds by showing that the laws of the traces of  $x = 10^n 10^{n+1} 10^{n+k}$  and  $y = 10^n 10^{n+k} 10^{n+1}$  are close to each other in the sense of Hellinger distance and concluding by a lemma in [21] that was used in a similar fashion to show a lower bound for linear trace reconstruction. It is first shown that conditioned on a 1 being deleted, a trace from  $x$  is equidistributed as a trace from  $y$ . Then explicit expressions for the probabilities that the trace is  $10^a 10^b 10^c$  are computed and compared, yielding an upper bound on the Hellinger distance. The difference between the probabilities for  $x$  and  $y$  is proportional to the product of  $(a - b)(b - c)(a - c)$  and a symmetric polynomial in  $a, b, c$ . Both  $x$  and  $y$  consist of three 1’s separated by runs of 0’s of approximate length  $n$ , so with high probability we have that  $a, b, c$  are approximately  $np$ , with square root fluctuations. The contribution of the  $(a - b)(b - c)(a - c)$  term allows us to recover a  $\tilde{\Omega}(n^3)$  bound.

## 1.4 Outline

In Section 2, we go over some preliminary definitions and results. In Section 3, we prove Theorem 2. In Section 4, we prove Theorem 4. In Section 5, we prove Theorem 6. Finally, in Section 6, we conclude by discussing open problems and avenues for further research. Some proofs, such as the proof of Proposition 1 and the full proof of Theorem 3, are deferred to Appendix A in the full version of this paper on arXiv.

## 2 Preliminaries

First, we explain a basic definition we will use involving complex numbers.

► **Definition 7.** For  $z \in \mathbb{C}$ , let  $|z|$  be the magnitude of  $z$ , and if  $z \neq 0$ , let  $\arg z$  be the argument of  $z$ , which is the value of  $\theta \in (-\pi, \pi]$  such that  $\frac{z}{|z|} = e^{i\theta}$ .

Next, we state a Littlewood-type result about bounding polynomials on arcs of the unit circle.

► **Theorem 8 ([9]).** Let  $f(z) = \sum_{j=0}^n a_j z^j$  be a nonzero polynomial of degree  $n$  with complex coefficients. Suppose there is some positive integer  $M$  such that  $|a_0| \geq 1$  and  $|a_j| \leq M$  for all  $0 \leq j \leq n$ . Then, if  $A$  is an arc of the unit circle  $\{z \in \mathbb{C} : |z| = 1\}$  with length  $0 < a < 2\pi$ , there exists some absolute constant  $c_1 > 0$  such that

$$\sup_{z \in A} |f(z)| \geq \exp\left(\frac{-c_1(1 + \log M)}{a}\right).$$

Next, we state two well known results about roots of unity in cyclotomic fields.

► **Lemma 9 ([29]).** Let  $\omega = e^{2\pi i/n}$ . Then, the set of  $\{\omega^k\}$  for  $k \in \mathbb{Z}, \gcd(k, n) = 1$  are all Galois conjugates. This means that if  $P(x)$  is an integer polynomial, then  $P(\omega^k) = 0$  if and only if  $P(\omega) = 0$  for any  $k \in \mathbb{Z}$  with  $\gcd(k, n) = 1$ . Moreover,  $P(\omega) = 0$  if and only if  $P$  is a multiple of the  $n$ th Cyclotomic polynomial.

► **Lemma 10 ([29]).** Let  $\omega = e^{2\pi i/n}$  be an  $n$ th root of unity, and let  $\mathbb{Q}[\omega]$  be the  $n$ th degree cyclotomic field. Then, if  $z \in \mathbb{Q}[\omega]$  is such that  $z^r = 1$  for some integer  $r \geq 1$ ,  $z$  must equal  $\omega^k$  or  $-\omega^k$  for some integer  $k$ .

Finally, we define the Hellinger distance between two probability measures and state a folklore bound on distinguishing between distributions based on samples in terms of the Hellinger distance.

► **Definition 11.** Let  $\mu$  and  $\nu$  be discrete probability measures over some set  $\Omega$ . In other words, for  $x \in \Omega$ ,  $\mu(x)$  is the probability of selecting  $x$  when drawing from the measure  $\mu$ . Then, the Hellinger distance is defined as

$$d_H(\mu, \nu) = \left( \sum_{x \in \Omega} \left( \sqrt{\mu(x)} - \sqrt{\nu(x)} \right)^2 \right)^{1/2}.$$

The following proposition is quite well-known (see for instance, [21, Lemma A.5]).

► **Proposition 12.** If  $\mu, \nu$  are discrete probability measures, then if given i.i.d. samples from either  $\mu$  or  $\nu$ , one must see at least  $\Omega(d_H(\mu, \nu)^{-2})$  i.i.d. samples to determine whether the distribution is  $\mu$  or  $\nu$  with at least  $2/3$  success probability.



### 3 Worst Case: Upper Bound

In this section, we prove Theorem 2, i.e., we provide an  $\exp(\tilde{O}(n^{1/3}))$ -sample algorithm for circular trace reconstruction when the length  $n$  is a prime or product of two primes.

For a (linear) string  $x \in \{0, 1\}^n$  and  $z \in \mathbb{C}$ , we define  $P(z; x) := \sum_{i=1}^n x_i z^i$ . The first lemma we require creates an unbiased estimator for  $\prod_{i=1}^m P(z_i; x)$  for some complex numbers  $z_1, \dots, z_m$ , using only random traces of  $x$ . The proof of the following lemma greatly resembles the proof of [31, Lemma 4.1], so we defer the proof to Appendix A in the full version of this paper on arXiv.

► **Lemma 13.** *Let  $x$  be a linear string of length  $n$ . Fix  $q$  as the deletion probability and  $p = 1 - q$  as the retention probability. Then, for any integer  $m \geq 1$  and any  $Z = (z_1, \dots, z_m)$  for  $z_1, \dots, z_m \in \mathbb{C}$ , there exists some function  $g_m(\tilde{x}, Z)$  such that*

$$\mathbb{E}_{\tilde{x}}[g_m(\tilde{x}, Z)] = \prod_{k=1}^m \left( \sum_{i=1}^n x_i z_k^i \right),$$

where the expectation is over traces drawn from  $x$ . Moreover, for any  $L \geq 1$ , and for all  $\tilde{x} \in \{0, 1\}^n$  and all  $Z$  such that  $|z_1|, \dots, |z_m| = 1$  and  $|\arg z_i| \leq \frac{1}{L}$  for all  $1 \leq i \leq m$ ,

$$|g_m(\tilde{x}, Z)| \leq (p^{-1}mn)^{O(m)} \cdot e^{O(m^2 n / (p^2 L^2))}.$$

For  $x \in \{0, 1\}^n$  and  $z \in \mathbb{C}$ , let  $P(z; x) := \sum_{i=1}^n x_i z^i$ . Our main goal will be to determine the value of  $f_t(z; x) := P(z; x)^t \cdot P(z^{-t}; x)$  for some integer  $t$ , where  $z$  is an  $n$ th root of unity. Importantly, we note that  $f_t(z; x)$  is invariant under rotations of  $x$ , since for  $z = e^{2\pi i k/n}$ ,

$$\sum_{i=1}^n x_{(i+1) \bmod n} z^i = \sum_{i=1}^n x_i z^{i-1} = P(z; x) \cdot z^{-1}$$

whereas

$$\sum_{i=1}^n x_{(i+1) \bmod n} z^{-t \cdot i} = \sum_{i=1}^n x_i z^{-t(i-1)} = P(z^{-t}; x) \cdot z^t.$$

Therefore, if we define  $x^{(j)}$  as the string  $x$  rotated by  $j$  places (so  $x_i^{(j)} = x_{(i+j) \bmod n}$ ), then  $f(z; x) = f(z; x^{(j)})$  for all  $z = e^{2\pi i k/n}$  and  $0 \leq j \leq n-1$ .

Now, choose some  $z$  with  $|z| = 1$  and  $|\arg z| \leq \frac{1}{L}$ . Also, fix some integer  $t$ , let  $m = t+1$ , and let  $Z = (\underbrace{z, \dots, z}_{t \text{ times}}, z^{-t})$ . Then, if  $j$  is randomly chosen in  $\{0, 1, \dots, n-1\}$  and  $\tilde{x}$  is a random trace,

$$\begin{aligned} \mathbb{E}_{\tilde{x}}[n z^{tn} \cdot g_m(\tilde{x}, Z)] &= (n \cdot z^{tn}) \cdot \left( \frac{1}{n} \cdot \sum_{j=0}^{n-1} P(z; x^{(j)})^t \cdot P(z^{-t}; x^{(j)}) \right) \\ &= \sum_{j=0}^{n-1} z^{tn} \cdot P(z; x^{(j)})^t \cdot P(z^{-t}; x^{(j)}), \end{aligned}$$

where  $g_m(\tilde{x}, Z)$  is defined in Lemma 13. Note that  $\sum_{j=0}^{n-1} z^{tn} \cdot P(z; x^{(j)})^t \cdot P(z^{-t}; x^{(j)})$  is a polynomial of  $z$  of degree at most  $3tn$  and all coefficients bounded by  $n^{t+1}$ . We write this polynomial as  $Q_t(z; x)$ . Thus, if we define  $h_t(\tilde{x}, z) := n z^{tn} g_m(\tilde{x}, Z)$ , we have that  $\mathbb{E}_{\tilde{x}}[h_t(\tilde{x}, z)] = Q_t(z; x)$  for  $\tilde{x}$  a trace of a randomly shifted  $x$ , and that  $|h_t(\tilde{x}; z)| \leq (p^{-1}tn)^{O(t)} \cdot e^{O(t^2 n / (p^2 L^2))}$  whenever  $|z| = 1$  and  $|\arg z| \leq \frac{1}{L}$  for  $L \geq 2$  and  $m = t+1$ , by Lemma 13.

Now, we will state two important results that will lead to the proof of the main result.

► **Lemma 14.** *Let  $n \geq 2$ , and suppose that  $x, x'$  are strings in  $\{0, 1\}^n$  such that  $Q_t(z; x) \neq Q_t(z; x')$  as polynomials in  $z$ . Then, there is an absolute constant  $c_2$  such that for any  $L \geq 2$ , there exists  $z$  such that  $|z| = 1$ ,  $|\arg z| \leq \frac{1}{L}$ , and*

$$|Q_t(z; x) - Q_t(z; x')| \geq n^{-c_2 t L}.$$

**Proof.** Note that  $Q_t(z; x) - Q_t(z; x')$  is a nonzero polynomial in  $z$  of degree at most  $(t+1)n$  and with all coefficients bounded by  $2n^{t+1}$ . Therefore, by Theorem 8,

$$\begin{aligned} \sup_{|z|=1, |\arg z| \leq 1/L} |Q_t(z; x) - Q_t(z; x')| &\geq \exp\left(-\frac{c_1(1 + \log(2n^{t+1}))}{2/L}\right) \\ &\geq \exp(-c_2 \cdot L \cdot t \cdot \log n) \\ &= n^{-c_2 t L}, \end{aligned}$$

where we note that the arc  $\{z : |z| = 1, |\arg z| \leq \frac{1}{L}\}$  has length  $\frac{2}{L}$ . ◀

The next important result we need will be Theorem 3. We defer the full proof of Theorem 3 to the full version of this paper on arXiv, but as the proof of the case where  $n$  is prime is simpler, we prove this special case here. Using this, we can get an  $\exp(\tilde{O}(n^{1/3}))$  sample upper bound at least for  $n$  prime. As a note, we will define  $\omega = e^{2\pi i/n}$  from now on, where  $n$  will be clear from context.

► **Proposition 15.** *Suppose that  $n = p$  is prime, and  $a_0, \dots, a_{p-1}, b_0, \dots, b_{p-1} \in \{0, 1\}$  are such that for all  $0 \leq k < p$ , there is some integer  $c_k$  such that  $\sum_{i=0}^{p-1} a_i = \omega^{c_k} \cdot \sum_{i=0}^{p-1} b_i$ . Then, the sequences  $\{a_1, \dots, a_p\}$  and  $\{b_1, \dots, b_p\}$  are equivalent up to a cyclic permutation.*

**Proof.** First,  $\sum_{i=0}^{p-1} a_i = \omega^{c_0} \cdot \sum_{i=0}^{p-1} b_i$ . Since  $\sum_{i=0}^{p-1} a_i$  and  $\sum_{i=0}^{p-1} b_i$  are both nonnegative real numbers, and since  $\omega^{c_0}$  is a root of unity, we must have that  $\sum_{i=0}^{p-1} a_i = \sum_{i=0}^{p-1} b_i$ .

Next, we have that  $\sum_{i=0}^{p-1} a_i \omega^i = \omega^{c_1} \cdot \sum_{i=0}^{p-1} b_i \omega^i$ . Letting  $b'_i = b_{(i-c_1) \pmod p}$ , we have that  $b'$  is a cyclic shift of  $b$ , and  $\sum_{i=0}^{p-1} a_i = \sum_{i=0}^{p-1} b'_i$  and  $\sum_{i=0}^{p-1} a_i \omega^i = \sum_{i=0}^{p-1} b'_i \omega^i$ . Letting  $Q(x) = \sum_{i=0}^{p-1} (a_i - b'_i) x^i$ , we have that  $\omega$  and 1 are both roots of  $Q(x)$ . Since  $Q(x)$  is an integer-valued polynomial, this implies that all Galois conjugates of  $\omega$  are roots, so  $1, \omega, \omega^2, \dots, \omega^{p-1}$  are roots of  $Q(x)$ . Thus,  $x^p - 1$  divides  $Q(x)$ . But since  $Q(x)$  has degree at most  $p-1$ ,  $Q(x)$  must equal 0, so  $a_i = b'_i$  for all  $i$ . Since the sequence  $b'$  is just a shift of  $b$ , we are done. ◀

By using Theorem 3 (or Proposition 15 in the case of  $n$  prime), we obtain the following number theoretic result.

► **Lemma 16.** *Let  $n$  be a prime or a product of two primes, and let  $a = a_1 a_2 \dots a_n$  and  $b = b_1 b_2 \dots b_n$  be distinct  $n$ -bit strings (even up to cyclic shift). Then, for some  $0 \leq \ell \leq n-1$  with  $z = \omega^\ell$ , and for some  $2 \leq t \leq 5$ , we have that  $P(z; a)^t P(z^{-t}; a) \neq P(z; b)^t P(z^{-t}; b)$ .*

**Proof.** First choose  $k$  such that  $\sum_{i=1}^n a_i \omega^{i \cdot k} \neq \omega^{c_k} \cdot \sum_{i=1}^n b_i \omega^{i \cdot k}$  for all integers  $c_k$ , which exists by Theorem 3. If  $k = 0$ , then  $P(\omega^k; a) = P(1; a)$  and  $P(\omega^k; b) = P(1; b)$  are distinct nonnegative integers, so we trivially have  $P(1; a)^t P(1; a) \neq P(1; b)^t P(1; b)$ . Otherwise, let  $t$  be the smallest prime that doesn't divide  $\frac{n}{\gcd(n, k)}$  (so  $t \leq 5$  as  $n$  has at most 2 prime factors). If  $\sum_{i=1}^n a_i \omega^{i \cdot k} = 0$ , then  $\sum_{i=1}^n b_i \omega^{i \cdot k} \neq 0$ . Now, since  $\omega^{-tk}$  is a Galois conjugate of  $\omega^k$  (since  $t \nmid n$ ), we also have that  $\sum_{i=1}^n b_i \omega^{-ti \cdot k} \neq 0$ . This means that  $P(\omega^k; a) = 0$  so  $P(\omega^k; a)^t P((\omega^k)^{-t}; a) = 0$ , but  $P(\omega^k; b)^t P((\omega^k)^{-t}; b) \neq 0$ . Likewise, if  $\sum_{i=1}^n b_i \omega^{i \cdot k} = 0$ , we'll have  $P(\omega^k; a)^t P((\omega^k)^{-t}; a) \neq 0$ , but  $P(\omega^k; b)^t P((\omega^k)^{-t}; b) = 0$ . This means the result follows if either  $P(\omega^k; a) = 0$  or  $P(\omega^k; b) = 0$ .

## 18:10 Circular Trace Reconstruction

Otherwise,  $P(\omega^k; a) = \sum_{i=1}^n a_i \omega^{i \cdot k}$  and  $P(\omega^k; b) = \sum_{i=1}^n b_i \omega^{i \cdot k}$  are both nonzero. This means that for all  $r \geq 0$ ,  $P(\omega^{(-t)^r \cdot k}; a)$  and  $P(\omega^{(-t)^r \cdot k}; b)$  are both nonzero, since  $\omega^{(-t)^r \cdot k}$  and  $\omega^k$  are Galois conjugates. This means that if  $P(z; a)^t P(z^{-t}; a) = P(z; b)^t P(z^{-t}; b)$  for all  $z = \omega^{(-t)^r \cdot k}$ , then

$$\frac{P(\omega^{(-t)^{r+1} \cdot k}; a)}{P(\omega^{(-t)^r \cdot k}; a)^{-t}} = \frac{P(z^{-t}; a)}{P(z; a)^{-t}} = \frac{P(z^{-t}; b)}{P(z; b)^{-t}} = \frac{P(\omega^{(-t)^{r+1} \cdot k}; b)}{P(\omega^{(-t)^r \cdot k}; b)^{-t}}$$

for all  $r \geq 0$ , so we inductively have that

$$\frac{P(\omega^{(-t)^r \cdot k}; a)}{P(\omega^k; a)^{(-t)^r}} = \frac{P(\omega^{(-t)^r \cdot k}; b)}{P(\omega^k; b)^{(-t)^r}}.$$

Now, letting  $r = \varphi\left(\frac{n}{\gcd(n, k)}\right)$ , we know that  $k \cdot (-t)^r \equiv k \pmod{n}$  by Euler's theorem, which means that  $\omega^{(-t)^r \cdot k} = \omega^k$ . Thus,

$$P(\omega^k; a)^{1-(-t)^r} = P(\omega^k; b)^{1-(-t)^r}.$$

Since  $k \neq 0$ , we have that  $\frac{n}{\gcd(n, k)} > 1$  so  $r \geq 1$ . Thus, since  $t \geq 2$ ,  $1 - (-t)^r \neq 0$ . Now, since  $P(\omega^k; a), P(\omega^k; b)$  are nonzero, we have that  $\frac{P(\omega^k; a)}{P(\omega^k; b)}$  is a  $|1 - (-t)^r|^{\text{th}}$  root of unity. Also,  $P(\omega^k; a), P(\omega^k; b) \in \mathbb{Q}[\omega]$ , which means  $\frac{P(\omega^k; a)}{P(\omega^k; b)} \in \mathbb{Q}[\omega]$ . However, all roots of unity in  $\mathbb{Q}[\omega]$  are of the form  $\pm \omega^i$  for some  $i$ , and since  $(-t)^r - 1$  is odd if  $n$  is odd (since  $t = 2$ ), we must have that  $\frac{P(\omega^k; a)}{P(\omega^k; b)} = \omega^{c_k}$  for some integer  $c_k$ . This is a contradiction, so we must have that  $P(z; a)^t P(z^{-t}; a) \neq P(z; b)^t P(z^{-t}; b)$ , for some  $z = \omega^{(-t)^r \cdot k}$ ,  $r \geq 0$ .  $\blacktriangleleft$

Finally, we are ready to prove Theorem 2.

**Proof of Theorem 2.** Suppose that we are trying to distinguish between the original circular string being  $a = a_1 a_2 \dots a_n$  or  $b = b_1 b_2 \dots b_n$ , where  $a, b$  are distinct, even up to cyclic shifts.

We choose  $\ell, t$  based on Lemma 16, so that  $P(\omega^\ell; a)^t P((\omega^\ell)^{-t}; a) \neq P(\omega^\ell; b)^t P((\omega^\ell)^{-t}; b)$ . As we have already noted, if  $z$  is an  $n^{\text{th}}$  root of unity, then  $P(z; a)^t P(z^{-t}; a)$  is invariant under rotation of  $a$ , and  $P(z; b)^t P(z^{-t}; b)$  is invariant under rotation of  $b$ . By our definition of  $Q_t(z; x)$ , we have that  $Q_t(\omega^\ell; a) \neq Q_t(\omega^\ell; b)$ , so  $Q_t(z; a) \neq Q_t(z; b)$  as polynomials in  $z$ . Therefore, by Lemma 14, there is some  $z$  such that  $|z| = 1, |\arg z| \leq \frac{1}{L}$ , and

$$|Q_t(z; a) - Q_t(z; b)| \geq n^{-c_2 t L} \geq n^{-5c_2 L}.$$

So, for  $L = \lceil n^{1/3} (\log n)^{-1/3} p^{-2/3} \rceil$ , there exists  $z$  with  $|z| = 1$  and  $|\arg z| \leq \frac{1}{L}$  and some  $2 \leq t \leq 5$  such that

$$|Q_t(z; a) - Q_t(z; b)| \geq n^{-5c_2 L} \geq \exp\left(-c_3 \cdot n^{1/3} (\log n)^{2/3} p^{-2/3}\right),$$

but

$$|h_t(\tilde{x}, z)| \leq (p^{-1} n)^{O(1)} \cdot \exp\left(O\left(\frac{n}{p^2 L^2}\right)\right) \leq \exp\left(c_4 \cdot n^{1/3} (\log n)^{2/3} p^{-2/3}\right)$$

for any trace  $\tilde{x}$  of either  $a$  or  $b$ . Sample  $R = \exp\left(O\left(n^{1/3} (\log n)^{2/3} p^{-2/3}\right)\right)$  traces  $\tilde{x}^{(1)}, \dots, \tilde{x}^{(R)}$  and choose  $z$  and  $t$  based on Lemma 16. Then, if we define  $h_t(z)$  to be the average of  $h_t(\tilde{x}^{(i)}, z)$  over  $i$  from 1 to  $R$ , the Chernoff bound tells us that with failure probability at most  $10^{-n}$ ,  $|h_t(z) - Q_t(z; a)| \leq \frac{1}{3} \cdot \exp\left(c_4 \cdot n^{1/3} (\log n)^{2/3} p^{-2/3}\right)$  if the original string were  $a$ ,

and  $|h(z) - Q_t(z; b)| \leq \frac{1}{3} \cdot \exp(c_4 \cdot n^{1/3}(\log n)^{2/3}p^{-2/3})$  if the original string were  $b$ . Thus, by returning  $a$  if  $h(z)$  is closer to  $Q_t(z; a)$  and returning  $b$  otherwise, we can distinguish between the original string being  $a$  or  $b$  using  $\exp(O(n^{1/3}(\log n)^{2/3}p^{-2/3}))$  traces, with  $1 - 10^{-n}$  failure probability.

Thus, to reconstruct the original string  $x$ , we simply run the distinguishing algorithm for all pairs  $a, b \in \{0, 1\}^n$  such that  $a \neq b$ , using the same  $R$  traces  $\tilde{x}^1, \dots, \tilde{x}^R$ . With probability at least  $1 - (4/10)^n \geq 1 - 2^{-n}$ , the true string  $x$  will be the only string such that the distinguishing algorithm will successfully choose  $x$  over all other strings. Thus, for  $n$  a prime or a product of two primes, the circular trace reconstruction problem can be solved using  $\exp(O(n^{1/3}(\log n)^{2/3}p^{-2/3}))$  traces. ◀

## 4 Average Case: Upper Bound

We now consider the situation in which the unknown circular string  $x$  is random. For the sake of simplicity, we will assume that we may sample  $x$  by sampling a uniform random linear binary string of length  $n$  and applying a uniform random cyclic shift. Note that the resulting distribution on  $x$  is not uniform over all possible circular strings, as strings with nontrivial cyclic symmetries are more likely to appear. However, such strings form a negligible fraction of all strings, and our arguments can easily be modified to handle the situation in which the distribution is uniform over all possible circular strings. We use the randomness to rule out certain problematic strings with high probability, and this can be done for uniform random circular strings as well as other distributions, for example if in the sampling procedure each bit is independently biased towards 0 or 1.

► **Theorem 17.** *Let  $x$  be a random (in the sense described above) unknown circular string of length  $n$  and let  $q$  be the deletion probability of each element. Then there exists a constant  $C_q$  depending only on  $q$  such that we can determine  $x$  with failure probability at most  $n^{-10}$  using  $O(n^{C_q})$  traces.*

In what follows, we will let  $x = x_1 \cdots x_n$  and take indices of bits in  $x$  modulo  $n$ . Let  $k = 100 \log n$ . We first note that with high probability, all of the consecutive substrings of  $x$  of length  $k$  and  $k - 1$  are pairwise distinct. We will refer to such strings  $x$  as *regular* strings. Indeed, the probability that  $x_i \cdots x_{i+k-1} = x_j \cdots x_{j+k-1}$  for  $i \neq j$  is  $2^{-k}$  (where indices are taken modulo  $n$ ), and union bounding over all  $i, j$  as well as both  $k$  and  $k - 1$  gives a failure probability of at most  $O(n^2 2^{-k}) \ll n^{-10}$ .

If we assume that  $x$  is regular, the length  $k$  consecutive substrings of  $x$  uniquely determine  $x$ . Indeed, given  $x_i \cdots x_{i+k-1}$ , we can uniquely determine  $x_{i+k}$  as there is a unique length  $k$  consecutive substring of  $x$  that begins with  $x_{i+1} \cdots x_{i+k-1}$ . Iteratively applying this allows us to recover the entire string  $x$ . Thus, to prove Theorem 17, it suffices to prove Lemma 5, i.e., to determine how many times each length  $k$  substring appears consecutively in any string  $x$  using  $O(n^{C_q})$  traces, which will allow us to recover  $x$  if  $x$  is regular.

**Proof of Lemma 5.** For a circular string  $x$ , let  $S_x = \{x_i x_{i+1} \cdots x_{i+k-1}\}_{i=1}^n$  be the  $k$ -deck of  $x$ , which is the multiset of contiguous substrings of  $x$  of length  $k$ . Let  $S$  and  $T$  be the  $k$ -decks of some circular strings of length  $n$  such that  $S \neq T$ . Suppose that given  $O(n^{C_q})$  traces of a circular string, we are able to distinguish between whether its  $k$ -deck is  $S$  or  $T$  correctly with failure probability  $10^{-n}$ . Then, by union bounding over all possible pairs of distinct  $k$ -decks (note that there are at most  $2^n$  different  $k$ -decks), we can with high probability correctly determine the  $k$ -deck of the string, showing the lemma.

The key property we will use is that given two distinct  $k$ -decks  $S$  and  $T$  that come from circular strings  $x$  and  $y$ , there exists some string  $s$  of length  $k$  such that the number of consecutive occurrences of  $s$  in  $x$  and in  $y$  are different. We will show the existence of  $C_q$  so that for any string  $s$  of length  $k$  satisfying this property, we can distinguish between  $x$  and  $y$  correctly using  $O(n^{C_q})$  samples with failure probability  $10^{-n}$ , from which the result follows.

Let  $\alpha$  denote a sufficiently large positive integer only depending on  $q$  that we will determine later. For  $0 \leq i \leq n-k$ , let  $c_i$  denote the number of (not necessarily consecutive) occurrences of  $s$  in  $x$  contained in a consecutive substring of  $x$  of length at most  $i+k$ . Similarly, let  $d_i$  denote the number of (not necessarily consecutive) occurrences of  $s$  in  $y$  contained in a consecutive substring of  $y$  of length at most  $i+k$ . By assumption, we have that  $c_0 \neq d_0$ . By casework on the last bit of the occurrence of  $s$ , we have that  $c_i, d_i \leq n \binom{i+k}{k}$ . Let  $P(t) = \sum_{i=0}^{\alpha k} c_i t^i$  and  $Q(t) = \sum_{i=0}^{\alpha k} d_i t^i$ . Moreover, the following is true:

► **Lemma 18.** *The probability that a trace of  $x$  starts with  $s$  (where a random bit in the string is chosen as the beginning before bits are deleted) is  $\frac{1}{n}(1-q)^k P(q) + O(q^{\alpha k}(\alpha+1)^k e^k)$ . Similarly, the probability that a trace of  $y$  starts with  $s$  is  $\frac{1}{n}(1-q)^k Q(q) + O(q^{\alpha k}(\alpha+1)^k e^k)$ .*

**Proof.** To compute the probability that a trace of  $x$  starts with  $s$ , we do casework on how many bits are deleted before the last bit in the occurrence of  $s$ . If  $i$  bits are deleted, then note that there are  $c_i$  ways for it to be done by definition. Each such way has a probability of  $\frac{1}{n}(1-q)^k q^i$  to occur. Indeed, for each way there is a  $\frac{1}{n}$  probability that the correct starting bit is chosen, and the probability that only the bits corresponding to the specific instance of  $s$  are kept is  $(1-q)^k q^i$ . It follows that the probability is exactly  $\frac{1}{n}(1-q)^k \sum_{i=0}^{n-k} c_i q^i$ .

It remains to show that  $\frac{1}{n}(1-q)^k \sum_{i=\alpha k}^{n-k} c_i q^i = O(q^{\alpha k}(\alpha+1)^k e^k)$ . As mentioned before, we have that  $c_i \leq n \binom{i+k}{k}$ . Thus, this sum is at most  $\sum_{i>\alpha k} \binom{i+k}{k} q^i \leq \binom{\alpha k+k}{k} q^{\alpha k} \sum_{i \geq 0} \left(\frac{q(\alpha+1)}{\alpha}\right)^i$ . Indeed, the ratio of consecutive terms in the sequence  $\binom{i+k}{k} q^i$  is equal to  $q \frac{i+k}{i} \leq \frac{q(\alpha+1)}{\alpha}$ . For a sufficiently large choice of  $\alpha$ ,  $\frac{q(\alpha+1)}{\alpha} < 1$ , so  $\sum_{i>\alpha k} \binom{i+k}{k} q^i = O\left(\binom{\alpha k+k}{k} q^{\alpha k}\right) = O(q^{\alpha k}(\alpha+1)^k e^k)$  by Stirling's approximation.

The argument for  $y$  is analogous. ◀

Lemma 18 allows us to estimate  $P(q)$  and  $Q(q)$  up to an  $O(n(1-q)^{-k} q^{\alpha k}(\alpha+1)^k e^k)$  error by looking at how often traces of  $x$  or  $y$  begin with  $s$ , and then dividing by  $\frac{1}{n}(1-q)^k$ . So long as  $P(q)$  and  $Q(q)$  are sufficiently far apart, a Chernoff bound allows us to determine with high probability if the traces came from  $x$  or  $y$ . However, it may be the case that  $P(q)$  and  $Q(q)$  are quite close. To remedy this, we observe that it is possible to *simulate higher deletion probabilities*  $q' > q$ . Indeed, this can be achieved by deleting each bit in traces received independently with probability  $\frac{q'-q}{1-q}$ . Thus, it suffices to find  $q' \in [q, r]$  with  $P(q')$  and  $Q(q')$  far apart for some  $q < r < 1$ . The existence of such a  $q'$  is proven by the following Littlewood-type result of Borwein, Erdélyi, and Kós.

► **Theorem 19** ([10], Theorem 5.1). *There exist absolute constants  $c_1 > 0$  and  $c_2 > 0$  such that if  $f$  is a polynomial with coefficients in  $[-1, 1]$  and  $a \in (0, 1]$ , then*

$$|f(0)|^{c_1/a} \leq \exp\left(\frac{c_2}{a}\right) \sup_{z \in [1-a, 1]} |f(z)|.$$

Let  $r = \frac{q+1}{2}$ . We first apply Theorem 19 to  $\binom{\alpha k+k}{k}^{-1}(P(rx) - Q(rx))$  and  $a = 1 - q/r$ . Here, we are using the fact that the coefficients of  $P$  and  $Q$  are bounded in magnitude by  $\binom{\alpha k+k}{k}$  by previous observations, and that  $|P(0) - Q(0)| \geq 1$ . Theorem 19 tells us that

$$\begin{aligned}
\binom{\alpha k + k}{k}^{-c_1/a} &\leq \exp\left(\frac{c_2}{a}\right) \binom{\alpha k + k}{k}^{-1} \sup_{z \in [1-a, 1]} |P(rz) - Q(rz)| \\
&= \exp\left(\frac{c_2}{a}\right) \binom{\alpha k + k}{k}^{-1} \sup_{q' \in [q, r]} |P(q') - Q(q')|,
\end{aligned}$$

or

$$\sup_{q' \in [q, r]} |P(q') - Q(q')| \geq c_3 \binom{\alpha k + k}{k}^{-c_4}$$

for some constants  $c_3$  and  $c_4$  that only depend on  $q$ .

In particular, this is much larger than  $10^k n(1-r)^{-k} r^{\alpha k} (\alpha+1)^k e^k$  for sufficiently large values of  $\alpha$  ( $\alpha$  may depend on  $q$ ). Indeed, after taking  $k$ th roots and using Stirling's approximation this reduces to showing that  $(e(\alpha+1))^{-c_5} > 10n^{1/k}(1-r)^{-1} r^\alpha (\alpha+1)e$  for sufficiently large  $\alpha$  where  $c_5$  is some constant that only depends on  $q$ , which is clear (since  $0 < r < 1$  is fixed and  $n^{1/k} < 2$ ). Thus, for any  $q' \in [q, r]$ , the error term  $\frac{1}{n}(1-q')^k \sum_{\alpha k+1}^{n-k} c_i(q')^i = O((q')^{\alpha k} (\alpha+1)^k e^k)$  is at most  $10^{-k}$  times  $\frac{1}{n}(1-q')^k \cdot \sup_{q' \in [q, r]} |P(q') - Q(q')|$ .

Hence, for some  $q' \in [q, r]$ , the probability that a trace begins with  $s$  under bit deletion with probability  $q'$  differs between  $x$  and  $y$  by  $\Omega(10^k n(1-r)^{-k} r^{\alpha k} (\alpha+1)^k e^k) = \Omega(n^{-c_6})$  for some constant  $c_6$  that only depends on  $q$ . By a standard Chernoff bound, for some constant  $C_q$  only depending on  $q$ , we can distinguish between  $x$  and  $y$  using  $O(n^{C_q})$  traces with failure probability at most  $\exp(-\Omega(n))$ , so Lemma 5 follows. ◀

By the previous discussions, Theorem 4 is also proven.

As mentioned before, Chen et. al. independently proved the analogue of Lemma 5 for linear strings in [15, Theorem 2]. The ideas behind their result and ours are similar: both are proved by considering a polynomial encoding the  $k$ -deck and using complex analysis to bound the corresponding polynomials for different  $k$ -decks away from each other on  $[q, 1]$ . Here, we directly apply the Littlewood-type result from [10], while Chen et. al. prove their own result for this bound. This idea of reconstructing the  $k$ -deck of the unknown string may be useful in other variants of trace reconstruction, though the sample complexity it achieves is only polynomial and in order to apply it, we must be able to reconstruct the string from its  $k$ -deck. For worst case trace reconstruction, we must be able to reconstruct strings not uniquely determined by their  $k$ -decks, and for average case trace reconstruction, a subpolynomial sample complexity has already been achieved for constant deletion probabilities. Thus, these ideas are not directly applicable to worst case and average case trace reconstruction, though they may be helpful for variants in which it suffices to construct  $k$ -decks and a polynomial sample complexity is not known, such as the ones considered in this paper and in [15].

One difference between our result and that of Chen et. al. is that while we only addressed the sample complexity, they also give a polynomial time algorithm for reconstructing the string via a linear program. Their approach can be modified to give a polynomial time algorithm for average case circular trace reconstruction as well. For more details on their algorithm, see [15, Section 6]. Moreover, while we do not address the smoothed complexity model as in Chen et. al., our proof of Theorem 4 easily generalizes to a polynomial sample (or time) algorithm for circular trace reconstruction in the smoothed complexity model. This is because one can show that a circular string drawn from Chen et. al.'s smoothed model is regular with very high probability, in a similar way to how we showed an average string is regular. For more details, see [15, Section 3].

## 5 Worst Case: Lower Bound

In this section, we prove Theorem 6 and demonstrate that worst-case circular trace reconstruction requires  $\tilde{\Omega}(n^3)$  traces. We first record the following lemma from [21] expressing the number of independent samples required to distinguish between two probability measures  $\mu$  and  $\nu$  in terms of their Hellinger distance  $d_H(\mu, \nu)$ , defined to be  $\left( \sum_{x \in X} \left( \sqrt{\mu(\{x\})} - \sqrt{\nu(\{x\})} \right)^2 \right)^{1/2}$  where the sum is over all events in some discrete sample space  $X$ . Let  $d_{TV}(\mu, \nu)$  denote the total variation distance between  $\mu$  and  $\nu$  and  $\mu^n$  denote the law of  $n$  independent samples from  $\mu$ .

► **Lemma 20** ([21], Lemma A.5). *If  $\mu$  and  $\nu$  are probability measures satisfying  $d_H(\mu, \nu) \leq 1/2$ , then  $1 - d_{TV}(\mu^m, \nu^m) \geq \varepsilon$  if  $m \leq \frac{\log(1/\varepsilon)}{9d_H^2(\mu, \nu)}$ .*

Given  $m$  traces, we cannot determine if they came from  $\mu^m$  or  $\nu^m$  with probability higher than  $\frac{1}{2}(1 + d_{TV}(\mu^m, \nu^m))$ . Thus, it requires  $\Omega(d_H^{-2}(\mu, \nu))$  samples to distinguish between two probability measures  $\mu$  and  $\nu$  with probability greater than  $\frac{3}{4}$ .

**Proof of Theorem 6.** We specialize to the case of distinguishing between  $x = 10^n 10^{n+1} 10^{n+k}$  and  $y = 10^n 10^{n+k} 10^{n+1}$  from independent traces. Let  $\mu$  and  $\nu$  respectively denote the laws of traces from  $x$  and  $y$ . We will show that  $d_H^2(\mu, \nu) = O((\log n/n)^3)$ , which establishes the result by Lemma 20.

First, we note that conditional on the first 1 in  $x$  being deleted, the resulting trace is equidistributed as a trace from  $y$  conditioned on the second 1 being deleted, as in both cases we obtain a trace from the circular string  $10^{n+1} 10^{2n+k}$ . Similar arguments for other cases show that conditioned on any 1 being deleted, traces from  $x$  and  $y$  are equal in law. Thus, the resulting string must have three 1's to contribute to the Hellinger distance. We will henceforth assume that the resulting trace is of the form  $10^a 10^b 10^c$  for some nonnegative integers  $a, b, c$ .

We now compute the ratio  $\frac{\mu(\{10^a 10^b 10^c\})}{\nu(\{10^a 10^b 10^c\})}$  and show that it is typically  $1 + O((\log n/n)^{3/2})$ . We have that

$$\begin{aligned} \frac{\mu(\{10^a 10^b 10^c\})}{q^{3n+k+1-a-b-c}(1-q)^{a+b+c}} &= \binom{n}{a} \binom{n+1}{b} \binom{n+k}{c} + \binom{n}{b} \binom{n+1}{c} \binom{n+k}{a} + \binom{n}{c} \binom{n+1}{a} \binom{n+k}{b}, \\ \frac{\nu(\{10^a 10^b 10^c\})}{q^{3n+k+1-a-b-c}(1-q)^{a+b+c}} &= \binom{n}{a} \binom{n+k}{b} \binom{n+1}{c} + \binom{n}{b} \binom{n+k}{c} \binom{n+1}{a} + \binom{n}{c} \binom{n+k}{a} \binom{n+1}{b}. \end{aligned}$$

It follows that

$$\frac{\mu(\{10^a 10^b 10^c\})}{\nu(\{10^a 10^b 10^c\})} = \frac{\frac{1}{(n+1-b)(n+1-c)\cdots(n+k-c)} + \frac{1}{(n+1-c)(n+1-a)\cdots(n+k-a)} + \frac{1}{(n+1-a)(n+1-b)\cdots(n+k-b)}}{\frac{1}{(n+1-c)(n+1-b)\cdots(n+k-b)} + \frac{1}{(n+1-a)(n+1-c)\cdots(n+k-c)} + \frac{1}{(n+1-b)(n+1-a)\cdots(n+k-a)}}.$$

Multiplying the numerator and denominator by  $\prod_{i=1}^k (n+i-a)(n+i-b)(n+i-c)$  results in

$$S_1 = \prod_{i=1}^k (n+i-a) \prod_{i=2}^k (n+i-b) + \prod_{i=1}^k (n+i-b) \prod_{i=2}^k (n+i-c) + \prod_{i=1}^k (n+i-c) \prod_{i=2}^k (n+i-a)$$

and

$$S_2 = \prod_{i=1}^k (n+i-b) \prod_{i=2}^k (n+i-a) + \prod_{i=1}^k (n+i-c) \prod_{i=2}^k (n+i-b) + \prod_{i=1}^k (n+i-a) \prod_{i=2}^k (n+i-c),$$



respectively. We have that  $S_1 - S_2 = (a - b) \prod_{i=2}^k (n + i - a)(n + i - b) + (b - c) \prod_{i=2}^k (n + i - b)(n + i - c) + (c - a) \prod_{i=2}^k (n + i - c)(n + i - a)$ . This is an alternating polynomial in  $a, b, c$ , i.e., applying a permutation  $\sigma$  to  $a, b, c$  changes the sign of the polynomial by the sign of  $\sigma$ . Hence, it can be written in the form  $(a - b)(b - c)(a - c)P_k(n, a, b, c)$ , where  $P_k$  is a polynomial in  $n, a, b, c$  of degree  $2k - 4$  since  $S_1$  and  $S_2$  have degree  $2k - 1$ .

By a standard Chernoff bound, there exists a constant  $C$  such that with probability at least  $1 - n^{-100}$ ,  $a, b, c \in [np - C\sqrt{n \log n}, np + C\sqrt{n \log n}]$ . When this occurs, we have that  $S_2 = \Omega(n^{2k-1})$  and  $|S_1 - S_2| = O((n \log n)^{3/2} n^{2k-4})$ , so  $\frac{\mu(\{10^a 10^b 10^c\})}{\nu(\{10^a 10^b 10^c\})} \in [1 - (c \log n/n)^{3/2}, 1 + (c \log n/n)^{3/2}]$  for some constant  $c$ . We thus have that

$$\begin{aligned} d_H^2(\mu, \nu) &= \sum_{a, b, c \geq 0} \left( \sqrt{\mu(\{10^a 10^b 10^c\})} - \sqrt{\nu(\{10^a 10^b 10^c\})} \right)^2 \\ &\leq 2n^{-100} + \sum_{a, b, c \in [np - C\sqrt{n \log n}, np + C\sqrt{n \log n}]} \nu(\{10^a 10^b 10^c\}) \left( 1 - \sqrt{\frac{\mu(\{10^a 10^b 10^c\})}{\nu(\{10^a 10^b 10^c\})}} \right)^2 \\ &= O((\log n/n)^3). \end{aligned}$$

It follows by Lemma 20 that it requires  $\Omega(n^3 / \log^3 n)$  samples to distinguish between traces from  $x$  and  $y$ , as desired.  $\blacktriangleleft$

## 6 Conclusion and Future Work

We note that our work leaves several open problems, including the following:

1. As noted in the introduction, Chase [14] very recently improved the worst-case linear trace reconstruction bound to  $\exp(\tilde{O}(n^{1/5}))$ . Is it possible to get a matching circular trace reconstruction bound, even just for certain lengths of strings?
2. For worst-case strings, can one get an upper bound of  $\exp(\tilde{O}(n^{1/3}))$  or even a subexponential bound for  $n$  with an arbitrary prime factorization?
3. Can one get a subpolynomial (i.e.,  $n^{o(1)}$ ) upper bound for the average case?
4. Can one improve our current lower bound for worst-case strings, perhaps even to  $n^{\omega(1)}$ ?
5. All of the work we have done in this paper has primarily focused on traces with constant deletion probability  $q$ . However, if  $q = o(1)$ , the implied results are no better than the bounds we get for fixed  $0 < q < 1$ . Can better bounds be obtained for circular trace reconstruction with small deletion probability (for instance,  $q = 1/(\log n)^2$ , or even  $q = n^{-2/3}$ )? In the linear case, there exist much better trace reconstruction algorithms in the low deletion probability regime (e.g., [7, 24]), so perhaps these results can be extended to circular strings.

For answering open problem 2, one method we attempted for getting a  $\exp(\tilde{O}(n^{1/2}))$  upper bound was to look at the polynomial  $P(y)P(z)P(y^{-1}z^{-1})$  for cyclotomic  $n$ th roots of unity  $y, z$ . One can establish a ‘‘Bivariate Littlewood’’-type result and the same argument as ours to show the following. Suppose that for any  $a, b \in \{0, 1\}^n$ ,  $P(y; a)P(z; a)P(y^{-1}z^{-1}; a) = P(y; b)P(z; b)P(y^{-1}z^{-1}; b)$  for all cyclotomic  $n$ th roots of unity  $y, z$  implies that  $a, b$  are equivalent up to cyclic rotation. Then, one can solve circular trace reconstruction using  $\exp(\tilde{O}(n^{1/2}))$  traces. This result may in fact look obvious, as if  $P(\omega_n; a) = \alpha \cdot P(\omega_n; b)$ , one should expect via a simple induction argument that  $P(\omega_n^k; a) = \alpha^k \cdot P(\omega_n^k; b)$  which implies that  $\alpha = e^{2\pi i r/n}$  for some  $r$  (by looking at  $k = n$ ). Thus, by rotating  $a$  by  $r$  elements, we will get that  $P(\omega_n^k; a) = P(\omega_n^k; b)$  for all  $k$ , which implies  $a = b$ . Unfortunately, one can have

cases where  $P(z; a) = P(z; b) = 0$  for several choices of  $z = e^{2\pi i k/n}$ , which can cause this induction argument to fail. Indeed, we believe that if such a result were true, proving it would again be a challenging number theoretic task.

We already noted some reasons in the introduction for why modifying the results of [35, 22] to answer open problem 3 is difficult. The main reason was that one cannot efficiently find the “start” of the string.

Finally, we note that a potential way of answering open problem 4 is via strings based on [21, 13]. One cannot use their strings directly, as their strings are equivalent up to a cyclic rotation, but perhaps an appropriate modification may improve upon our  $\tilde{\Omega}(n^3)$  lower bound.

---

## References

- 1 Circular DNA. Available at [https://en.wikipedia.org/wiki/Circular\\_DNA](https://en.wikipedia.org/wiki/Circular_DNA).
- 2 Alexandr Andoni, Assaf Goldberger, Andrew McGregor, and Ely Porat. Homomorphic fingerprints under misalignments: sketching edit and shift distances. In *Symposium on Theory of Computing (STOC)*, pages 931–940, 2013. doi:10.1145/2488608.2488726.
- 3 Frank Ban, Xi Chen, Adam Freilich, Rocco A. Servedio, and Sandip Sinha. Beyond trace reconstruction: Population recovery from the deletion channel. In *Foundations of Computer Science (FOCS)*, pages 745–768, 2019. doi:10.1109/FOCS.2019.00050.
- 4 Frank Ban, Xi Chen, Rocco A. Servedio, and Sandip Sinha. Efficient average-case population recovery in the presence of insertions and deletions. In *Approximation, Randomization, and Combinatorial Optimization: Algorithms and Techniques*, pages 44:1–44:18, 2019. doi:10.4230/LIPIcs.APPROX-RANDOM.2019.44.
- 5 Afonso S. Bandeira, Moses Charikar, Amit Singer, and Andy Zhu. Multireference alignment using semidefinite programming. In *Innovations in Theoretical Computer Science (ITCS)*, pages 745–768, 2019. doi:10.1145/2554797.2554839.
- 6 Afonso S. Bandeira, Jonathan Niles-Weed, and Philippe Rigollet. Optimal rates of estimation for multi-reference alignment. *Mathematical Statistics and Learning*, 2(1):25–75, 2019. doi:10.4171/MSL/11.
- 7 Tugkan Batu, Sampath Kannan, Sanjeev Khanna, and Andrew McGregor. Reconstructing strings from random traces. In *Symposium on Discrete Algorithms (SODA)*, pages 910–918, 2004. URL: <http://dl.acm.org/citation.cfm?id=982792.982929>.
- 8 Vinnu Bhardwaj, Pavel A. Pevzner, Cyrus Rashtchian, and Yana Safonova. Trace reconstruction problems in computational biology. *IEEE Transactions on Information Theory*, pages 1–1, 2020. doi:10.1109/TIT.2020.3030569.
- 9 Peter Borwein and Tamás Erdélyi. Littlewood-type polynomials on subarcs of the unit circle. *Indiana University Mathematics Journal*, 46(4):1323–1346, 1997. doi:10.1512/IUMJ.1997.46.1435.
- 10 Peter Borwein, Tamás Erdélyi, and Géza Kós. Littlewood-type problems on  $[0, 1]$ . *Proceedings of the London Mathematical Society*, 3(79):22–46, 1999. doi:10.1112/S0024611599011831.
- 11 Joshua Brakensiek, Ray Li, and Bruce Spang. Coded trace reconstruction in a constant number of traces. In *Foundations of Computer Science (FOCS)*, 2020. arXiv:1908.03996.
- 12 Panagiotis Charalampopoulos, Tomasz Kociumaka, Solon P. Pissis, Jakub Radoszewski, Wojciech Rytter, Juliusz Straszłyński, Tomasz Waleń, and Wiktor Zuba. Circular pattern matching with  $k$  mismatches. *Journal of Computer and System Sciences*, 115:73–85, 2021. doi:10.1016/j.jcss.2020.07.003.
- 13 Zachary Chase. New lower bounds for trace reconstruction. *CoRR*, abs/1905.03031, 2019. arXiv:1905.03031.
- 14 Zachary Chase. New upper bounds for trace reconstruction. *CoRR*, abs/2009.03296, 2020. arXiv:2009.03296.

- 15 Xi Chen, Anindya De, Chin Ho Lee, Rocco A. Servedio, and Sandip Sinha. Polynomial-time trace reconstruction in the smoothed complexity model. *CoRR*, abs/2008.12386, 2020. To appear in Symposium on Discrete Algorithms (SODA), 2021. [arXiv:2008.12386](#).
- 16 Mahdi Cheraghchi, Ryan Gabrys, Olga Milenkovic, and João Ribeiro. Coded trace reconstruction. *IEEE Trans. Inf. Theory*, 66(10):6084–6103, 2020. doi:10.1109/TIT.2020.2996377.
- 17 George M. Church, Yuan Gao, and Sriram Kosuri. Next-generation digital information storage in DNA. *Science*, 337(6102):1628, 2012. doi:10.1126/science.1226355.
- 18 Sami Davies, Miklos Racz, and Cyrus Rashtchian. Reconstructing trees from traces. In *Conference On Learning Theory (COLT)*, pages 961–978, 2019. URL: <http://proceedings.mlr.press/v99/davies19a.html>.
- 19 Anindya De, Ryan O'Donnell, and Rocco A. Servedio. Optimal mean-based algorithms for trace reconstruction. *Annals of Applied Probability*, 29(2):851–874, 2019. doi:10.1214/18-AAP1394.
- 20 Lisa Hartung, Nina Holden, and Yuval Peres. Trace reconstruction with varying deletion probabilities. In *Analytic Algorithmics and Combinatorics (ANALCO)*, pages 54–61, 2018. doi:10.1137/1.9781611975062.6.
- 21 Nina Holden and Russell Lyons. Lower bounds for trace reconstruction. *Annals of Applied Probability*, 30(2):503–525, 2020. doi:10.1214/19-AAP1506.
- 22 Nina Holden, Robin Pemantle, and Yuval Peres. Subpolynomial trace reconstruction for random strings and arbitrary deletion probability. In *Conference On Learning Theory (COLT)*, pages 1799–1840, 2018. URL: <http://proceedings.mlr.press/v75/holden18a.html>.
- 23 Thomas Holenstein, Michael Mitzenmacher, Rina Panigrahy, and Udi Wieder. Trace reconstruction with constant deletion probability and related results. In *Symposium on Discrete Algorithms (SODA)*, pages 389–398, 2008. doi:10.1145/1347082.1347125.
- 24 Sampath Kannan and Andrew McGregor. More on reconstructing strings from random traces: insertions and deletions. In *International Symposium on Information Theory (ISIT)*, pages 297–301, 2005. doi:10.1109/ISIT.2005.1523342.
- 25 Akshay Krishnamurthy, Arya Mazumdar, Andrew McGregor, and Soumyabrata Pal. Trace reconstruction: Generalized and parameterized. In *European Symposium on Algorithms (ESA)*, pages 68:1–68:25, 2019. doi:10.4230/LIPIcs.ESA.2019.68.
- 26 Vladimir I. Levenshtein. Efficient reconstruction of sequences. *IEEE Trans. Information Theory*, 47(1):2–22, 2001. doi:10.1109/18.904499.
- 27 Vladimir I. Levenshtein. Efficient reconstruction of sequences from their subsequences or supersequences. *J. Comb. Theory, Ser. A*, 93(2):310–332, 2001. doi:10.1006/jcta.2000.3081.
- 28 Maurice Maes. On a cyclic string-to-string correction problem. *Information Processing Letters*, 35(2):73–78, 1990. doi:10.1016/0020-0190(90)90109-B.
- 29 Daniel A. Marcus. *Number Fields*. Springer International Publishing, 1977.
- 30 Andrew McGregor, Eric Price, and Sofya Vorotnikova. Trace reconstruction revisited. In *European Symposium on Algorithms (ESA)*, pages 689–700, 2014.
- 31 Shyam Narayanan. Population recovery from the deletion channel: Nearly matching trace reconstruction bounds. *CoRR*, abs/2004.06828, 2020. To appear in Symposium on Discrete Algorithms (SODA), 2021. [arXiv:2004.06828](#).
- 32 Fedor Nazarov and Yuval Peres. Trace reconstruction with  $\exp(o(n^{1/3}))$  samples. In *Symposium on Theory of Computing (STOC)*, pages 1042–1046, 2017. doi:10.1145/3055399.3055494.
- 33 Hoang Hiep Nguyen, Jeho Park, Seon Joo Park, Chang-Soo Lee, Seungwoo Hwang, Yong-Beom Shin, Tai Hwan Ha, and Moonil Kim. Long-term stability and integrity of plasmid-based dna data storage. *Polymers*, 10(1):28, 2018. doi:10.3390/polym10010028.
- 34 Lee Organick, Siena Dumas Ang, Yuan-Jyue Chen, Randolph Lopez, Sergey Yekhanin, Konstantin Makarychev, Miklos Z Racz, Govinda Kamath, Parikshit Gopalan, Bichlien Nguyen, Christopher N Takahashi, Sharon Newman, Hsing-Yeh Parker, Cyrus Rashtchian, Kendall Stewart, Gagan Gupta, Robert Carlson, John Mulligan, Douglas Carmean, Georg Seelig, Luis Ceze, , and Karin Strauss. Random access in large-scale dna data storage. *Nature Biotechnology*, 36:242–248, 2018. doi:10.1038/nbt.4079.

- 35 Yuval Peres and Alex Zhai. Average-case reconstruction for the deletion channel: Subpolynomially many traces suffice. In *Foundations of Computer Science (FOCS)*, pages 228–239, 2017. doi:10.1109/FOCS.2017.29.
- 36 Amelia Perry, Jonathan Weed, Afonso S. Bandeira, Philippe Rigollet, and Amit Singer. The sample complexity of multireference alignment. *SIAM Journal on Mathematics of Data Science*, 1(3):497–517, 2019. doi:10.1137/18M1214317.
- 37 Jane B. Reece, Lisa A. Urry, Michael L. Cain, Steven A. Wasserman, Peter V. Minorsky, and Robert B. Jackson. *Campbell Biology*. Pearson, 9th edition, 2011.
- 38 Krishnamurthy Viswanathan and Ram Swaminathan. Improved string reconstruction over insertion-deletion channels. In *Symposium on Discrete Algorithms (SODA)*, pages 399–408, 2008. doi:10.1145/1347082.1347126.

# Self-Testing of a Single Quantum Device Under Computational Assumptions

Tony Metger 

Institute for Theoretical Physics, ETH Zürich, Switzerland  
tmetger@ethz.ch

Thomas Vidick 

Department of Computing and Mathematical Sciences,  
California Institute of Technology, Pasadena, CA, USA  
vidick@caltech.edu

---

## Abstract

Self-testing is a method to characterise an arbitrary quantum system based only on its classical input-output correlations, and plays an important role in device-independent quantum information processing as well as quantum complexity theory. Prior works on self-testing require the assumption that the system's state is shared among multiple parties that only perform local measurements and cannot communicate. Here, we replace the setting of *multiple non-communicating* parties, which is difficult to enforce in practice, by a *single computationally bounded* party. Specifically, we construct a protocol that allows a classical verifier to robustly certify that a single computationally bounded quantum device must have prepared a Bell pair and performed single-qubit measurements on it, up to a change of basis applied to both the device's state and measurements. This means that under computational assumptions, the verifier is able to certify the presence of entanglement, a property usually closely associated with two separated subsystems, inside a single quantum device. To achieve this, we build on techniques first introduced by Brakerski et al. (2018) and Mahadev (2018) which allow a classical verifier to constrain the actions of a quantum device assuming the device does not break post-quantum cryptography.

**2012 ACM Subject Classification** Theory of computation → Quantum computation theory

**Keywords and phrases** Quantum computing, quantum cryptography, device-independence, self-testing, post-quantum cryptography

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.19

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/2001.09161>.

**Funding** *Tony Metger*: supported by the ETH Foundation through the Excellence Scholarship & Opportunity Programme, and the IQIM, an NSF Physics Frontiers Center (NSF Grant PHY-1125565) with support of the Gordon and Betty Moore Foundation (GBMF-12500028).

*Thomas Vidick*: supported by NSF CAREER Grant CCF-1553477, AFOSR YIP award number FA9550-16-1-0495, a CIFAR Azrieli Global Scholar award, MURI Grant FA9550-18-1-0161, and the IQIM, an NSF Physics Frontiers Center (NSF Grant PHY-1125565) with support of the Gordon and Betty Moore Foundation (GBMF-12500028).

**Acknowledgements** We thank Andrea Coladangelo, Andru Gheorghiu, Anand Natarajan, and Tina Zhang for helpful discussions; Andrea Coladangelo, Andru Gheorghiu, Urmila Mahadev, and Akihiro Mizutani for comments on the manuscript; and Lídia del Río for pointing out the reference [6]. This work was carried out while Tony Metger was a visiting student researcher at the Department of Computing and Mathematical Sciences at Caltech.

## 1 Introduction

The *device-independent* approach to quantum information processing treats quantum devices as black boxes which we can interact with classically to observe their input-output correlations. Based solely on these correlations and the assumption that quantum mechanics is correct,



© Tony Metger and Thomas Vidick;  
licensed under Creative Commons License CC-BY  
12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 19; pp. 19:1–19:12



Leibniz International Proceedings in Informatics  
LIPIcs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

the goal is to prove statements about the devices, e.g., to show that they can be used for secure quantum key distribution (see e.g. [24]) or delegated quantum computation (see e.g. [32]). At a fundamental level, this provides a theory-of-computation approach to the study of classical signatures of quantum mechanics and their use as a “leash” to control and characterize quantum devices.

*Self-testing* is arguably the most effective method in device-independent quantum information processing. The goal in self-testing is to characterise the quantum state and measurements of multiple black-box quantum devices using only their classical input-output correlations. In analogy to the setting of interactive proof systems, the classical party observing the input-output correlations is sometimes called the *verifier*, and the black-box quantum devices are called *provers*. More specifically, the verifier can interact with multiple quantum provers by sending (classical) questions as inputs and receiving (classical) answers as outputs. The provers can share any (finite-dimensional) entangled quantum state at the start of the interaction and are computationally unbounded; however, it is assumed that after having received the verifier’s questions, the provers can no longer communicate. Based on the question-answer correlations, the verifier would like to deduce that the provers must have shared a certain initial state and performed certain measurements on it, up to a local change of basis on each prover’s Hilbert space. We will describe this scenario in more detail in Section 1.1. We emphasize that self-testing is a uniquely quantum phenomenon: for classical devices, there is simply a function that is implemented by the device, and it is not meaningful to ask *how* the function is implemented “on the inside”. In contrast, for quantum devices, in certain cases knowledge of the function (the observed input-output behaviour) implies an essentially unique realization in terms of a quantum state and measurements on it.

The term *self-testing* was introduced by Mayers and Yao in [24] in the context of proofs of security for quantum key distribution, but the notion was already present in earlier works [34, 29]. For a review covering a large number of different self-testing protocols, as well as applications such as randomness expansion and delegated quantum computation, see [35]. In addition to more practical applications, self-testing has also proved to be a powerful tool in quantum complexity theory for the study of multi-prover interactive proof systems in the quantum setting and is at the heart of the recent characterisation of the complexity class  $\text{MIP}^*$  [21].

The starting point for our work is the observation that, while the model of non-communicating quantum provers used in existing self-testing results is appealing in theory, it is difficult to enforce this non-communication assumption in practice. Motivated by the many applications of self-testing in quantum cryptography (e.g. device-independent quantum key distribution) and complexity theory, we are compelled to search for protocols that allow for a self-testing-like certification of a *single* untrusted quantum device.

Self-testing protocols in the multi-prover setting are typically based on the violation of Bell inequalities [3], for which the non-communication assumption is necessary.<sup>1</sup> Hence, different techniques or additional assumptions are necessary when considering the single-device scenario. What could a “computational Bell inequality” look like?

In this paper we give an answer to this question by constructing a self-testing protocol for a *single computationally bounded* quantum device. Specifically, the only assumptions required are the correctness of quantum mechanics and that the prover does not have the

---

<sup>1</sup> Another approach is to base the self-testing statement on non-contextuality inequalities [5, 6]. The violation of non-contextuality inequalities is a uniquely quantum phenomenon that is similar to the violation of Bell inequalities, with the advantage that it only requires a single quantum device and therefore no non-communication assumption. The downside of this approach is that it places additional assumptions, such as memory constraints and compatibility relations between measurements, on the quantum device, limiting its suitability for practical cryptographic applications.



ability to break the Learning with Errors (LWE) assumption [31], a common assumption in post-quantum cryptography, *during* the protocol execution (whereas breaking the LWE assumption *after* the end of the protocol is allowed). Our protocol is a three-round interaction between a classical verifier and a quantum prover, at the end of which the verifier decides to either “accept” or “reject” the prover. Informally, the guarantee provided by the protocol is the following:

► **Theorem (Informal).** *A prover’s strategy in the protocol is described by a quantum state and the measurements that the prover makes on the state to obtain the (classical) answers received by the verifier. If a computationally bounded prover is accepted by the verifier with probability  $1 - \varepsilon$ , then there exists an isometry  $V$  such that for a universal constant  $c > 0$  and under the isometry  $V$ :*

1. *the prover’s state is  $O(\varepsilon^c)$ -close (in trace distance) to a Bell pair,*
2. *(a subset of) the prover’s measurements are  $O(\varepsilon^c)$ -close to single-qubit measurements in the computational or Hadamard basis, where the measurement bases are chosen by the verifier. Here, “closeness” is measured in a distance measure suitable for measurements acting on a state.*

We emphasize that the theorem not only guarantees the preparation of an entangled state by the prover, but also the implementation of specific measurements on it. As such, it provides a complete analogue of foundational self-testing results for the CHSH inequality [34, 25].

The proof of our main result builds on techniques introduced in recent works [23, 8, 19] to allow a classical verifier to leverage post-quantum cryptography to control a computationally bounded quantum prover. Because they are relevant for understanding the proof of our results, we now give a brief overview of these works and explain their relation to self-testing.

In [23], Mahadev gives the first protocol to classically verify a delegated quantum computation with a single untrusted quantum prover. The central ingredient in Mahadev’s verification protocol is a “measurement protocol” that allows the verifier to force the prover to report classical outcomes obtained by performing certain measurements on a quantum state that the prover has “committed to” using classical information. The main guarantee of the measurement protocol is this: if the prover is accepted in the protocol, *there exists* a quantum state such that the distribution over the prover’s answers could have been produced by performing the requested measurements on this state. In other words, all of the prover’s answers must be self-consistent in the sense that they could have originated from performing different measurements on (copies of) the same quantum state.

To verify a quantum computation, the statement that the prover’s answers are *consistent* with measurements on a quantum state is sufficient, as the *existence* of a quantum state with the right properties can certify the outcome of the quantum computation (this is due to Kitaev’s “circuit-to-Hamiltonian” construction, which we do not explain here). However, in this work we seek to make a stronger statement: we want to certify that the prover *actually constructed* the desired quantum state *and performed the desired measurements* on it (up to an isometry). While the honest prover in Mahadev’s protocol does indeed construct the desired quantum state, the protocol does not guarantee that an arbitrary prover must do, too. Hence, our self-testing protocol is stronger in the sense that it allows for a more stringent characterisation of the prover’s actions, namely its actual states and measurements.<sup>2</sup> To emphasize the difference, we note that the guarantee of Mahadev’s protocol *does not* directly

<sup>2</sup> This comes at the cost that we are only able to certify Bell pairs, while Mahadev’s measurement protocol works for measurements on any state.



imply that a successful prover must have performed any quantum computation; the guarantee is only that, if the correct state preparation and measurements were to be performed, the outcome would be as claimed by the prover.

Another closely related work is that of Brakerski et al. [8], who give a protocol between a classical verifier and a quantum prover that allows the verifier to generate certified information-theoretic randomness, again assuming that the prover does not break the LWE assumption; in other words, their protocol generates information-theoretic randomness from a computational assumption. For this, the authors show that two of the prover’s measurements must be maximally incompatible, as defined by a quantity that they call the “overlap”. Informally, one can think of two maximally incompatible measurements as being close to a computational and Hadamard basis measurement, up to some global change of basis. Hence, this result already resembles self-testing in the sense that the verifier can make a statement about the actual measurements used by the prover. In particular, it does serve as a “test of quantumness” for the prover.

Building on [8] and using techniques from [23], Gheorghiu and Vidick construct a protocol for a task that they call verifiable remote state preparation (RSP) [19]. They consider a set of single-qubit pure states  $\{|\psi_1\rangle, \dots, |\psi_n\rangle\}$ .<sup>3</sup> Under the same LWE assumption as before, the protocol enables the verifier to certify that the prover has prepared one of these states, up to a global change of basis (i.e., some isometry  $V$  that is applied to all  $|\psi_i\rangle$ ). More precisely, the verifier cannot decide beforehand on a particular  $|\psi_i\rangle$ , but after executing the protocol, the verifier knows which  $|\psi_i\rangle$  the prover has prepared, and the distribution over  $i$  can be made uniform. The prover, on the other hand, does not know which  $|\psi_i\rangle$  he has prepared.

This result resembles a self-testing statement even more than that of [8] because it explicitly characterises a family of single-qubit quantum states, one of which is certified to be present in the prover’s space. However, it differs from a standard self-testing statement in that it is defined for a family of states, not an individual state: because the prover’s isometry  $V$  is arbitrary, any individual state  $|\psi_i\rangle$  can be mapped to another arbitrary state. Hence, what is certified in RSP is not any individual state, but the relationships (e.g., orthogonality) between different states in some family. Alternatively, one can also take the view that RSP characterises the relationships between the prover’s states and measurements. We return to this issue in more detail in Section 1.1. The idea of certifying a family of states has also been considered by Cojocaru et al. [12], who call this notion “blind self-testing”. They analyze a different protocol under a restricted adversarial model and conjecture that their protocol yields similar guarantees as [19] for single-qubit states and tensor products of single-qubit states.

This lengthy overview of previous works makes explicit a progression towards the task that we tackle here, that of genuine self-testing of a single quantum device. We note that this presentation clearly benefits from hindsight, and that none of the cited works mentions any relation to self-testing; indeed, the results are too weak to be used in this setting. In particular, none of the previous works provides a sufficiently strong guarantee on the measurements performed by the quantum device and goes beyond the setting of a single qubit, which is arguably the main technical challenge. Indeed, moving from a single-qubit state to an entangled two-qubit state means that the verifier has to enforce a tensor product structure on the prover’s space, which is one of the main difficulties in our soundness proof ([27, Section 4]). On a technical level, it requires the certification of compatibility relations

---

<sup>3</sup> The protocol in [19] is designed for a specific set of ten pure states that are useful for delegated quantum computation, but for the purposes of this overview it is not important which specific states these are.

between different measurements meant to act on different qubits. Additionally, having two qubits instead of one prevents us from using Jordan’s lemma, a standard tool in self-testing also used in [19], to characterise the prover’s measurements; in [27, Section 4.7], we show how to characterise the prover’s measurements using a different method starting with a *partial* characterisation of the prover’s measurements, using that to partially characterise the prover’s states, which in turn is used for a stronger partial characterisation of the measurements, etc., until we reach the full statement that shows that the prover makes single-qubit measurements on a Bell pair.

## 1.1 Self-testing in the multi- and single-prover settings

In this section, we give a brief overview of the standard multi-prover self-testing scenario, and explain how it can be extended to a single prover. For more details on the multi-prover scenario, see [35] or [33, Chapter 7]. For simplicity, let us consider the case of two provers  $A$  and  $B$ , with Hilbert spaces  $\mathcal{H}_A$  and  $\mathcal{H}_B$ , respectively. Hence, the total Hilbert space is  $\mathcal{H}_A \otimes \mathcal{H}_B$ . The verifier interacts with  $A$  and  $B$  by sending questions and receiving answers. The question-answer correlations can be described by a family of probability distributions  $\{p(a, b|x, y)\}_{x, y}$ , where for each choice of questions  $x$  and  $y$  sent to  $A$  and  $B$ , respectively,  $p(a, b|x, y)$  is a probability distribution over their answers  $a$  and  $b$ . We say that a quantum state  $|\psi\rangle_{AB} \in \mathcal{H}_A \otimes \mathcal{H}_B$  is *compatible* with the correlations  $p(a, b|x, y)$  if there are local measurements  $\{P_x^{(a)}\}_a$  on  $\mathcal{H}_A$  for every input  $x$ , and  $\{Q_y^{(b)}\}_b$  on  $\mathcal{H}_B$  for every input  $y$ , that realise the correlations  $p(a, b|x, y)$ , i.e.,  $p(a, b|x, y) = \langle \psi | P_x^{(a)} \otimes Q_y^{(b)} | \psi \rangle_{AB}$  for all  $x, y, a, b$ .

► **Definition 1** (Self-testing of states, informal). *The correlations  $p(a, b|x, y)$  self-test a state  $|\phi\rangle_{AB}$  if for any state  $|\psi\rangle_{AB}$  compatible with these correlations, there exists a local isometry  $V = V_A \otimes V_B$  (with  $V_A$  only acting on  $\mathcal{H}_A$ , and  $V_B$  only acting on  $\mathcal{H}_B$ ) such that  $V|\psi\rangle_{AB} = |\phi\rangle_{AB}|AUX\rangle$  for some ancillary state  $|AUX\rangle$ .*

A more operational view of this statement is that it must be possible to “extract” the state  $|\phi\rangle_{AB}$  from  $|\psi\rangle_{AB}$  only by performing local operations. The condition that the isometry must be local is crucial: if we would allow a global isometry, we could map any state  $|\psi\rangle_{AB}$  to the desired state  $|\phi\rangle_{AB}$ . In the two-prover case, the notion of a *local* isometry is natural, since the separation between the two provers induces a tensor product structure  $\mathcal{H} = \mathcal{H}_A \otimes \mathcal{H}_B$  on the global Hilbert space  $\mathcal{H}$ . In contrast, for a single prover no such tensor product structure exists and we cannot define *local* isometries in a meaningful way.

In Definition 1, we only dealt with the provers’ state, not his measurements. A stronger notion of self-testing is to characterise both the provers’ state and measurements. This is the version of self-testing originally considered by Mayers and Yao [24], and we will see that it can be meaningfully extended to the single-prover setting.

► **Definition 2** (Self-testing of states and measurements, informal). *The correlations  $p(a, b|x, y)$  self-test a state  $|\phi\rangle_{AB}$  and measurements  $\{M_x^{(a)}\}, \{N_y^{(b)}\}$  if for any state  $|\psi\rangle_{AB}$  and measurements  $\{P_x^{(a)}\}, \{Q_y^{(b)}\}$  that realise the correlations  $p(a, b|x, y)$ , there exists a local isometry  $V = V_A \otimes V_B$  such that*

1.  $V|\psi\rangle_{AB} = |\phi\rangle_{AB}|AUX\rangle$ ,
2.  $V(P_x^{(a)} \otimes Q_y^{(b)})|\psi\rangle_{AB} = ((M_x^{(a)} \otimes N_y^{(b)})|\phi\rangle_{AB})|AUX\rangle$ , for some ancillary state  $|AUX\rangle$ .

The first condition is the same as in Definition 1. The second condition roughly says that the “physical” measurements  $\{P_x^{(a)}\}$  and  $\{Q_y^{(b)}\}$  used by  $A$  and  $B$ , respectively, act on the state  $|\psi\rangle_{AB}$  in the same way that the desired measurements  $\{M_x^{(a)}\}$  and  $\{N_y^{(b)}\}$  act on the desired state  $|\phi\rangle_{AB}$ .

Self-testing of states and measurements still has meaning in the single-prover setting. In this setting, one can imagine that the verifier sends both questions  $x$  and  $y$  to the same prover, and the prover replies with two answers  $a$  and  $b$ . To compute his answers, the prover prepares a quantum state  $|\psi\rangle$  and, on inputs  $x, y$ , performs a measurement  $\{P_{x,y}^{(a,b)}\}_{a,b}$  to obtain answers  $a, b$ .

► **Definition 3** (Self-testing for a single prover, informal). *The correlations  $p(a, b|x, y)$  self-test a state  $|\phi\rangle$  and measurements  $\{K_{x,y}^{(a,b)}\}_{a,b}$  if for any state  $|\psi\rangle$  and measurements  $\{P_{x,y}^{(a,b)}\}_{a,b}$  that realise the correlations  $p(a, b|x, y)$ , there exists an isometry  $V$  such that*

1.  $V|\psi\rangle = |\phi\rangle|AUX\rangle$ ,
2.  $VP_{x,y}^{(a,b)}|\psi\rangle = \left(K_{x,y}^{(a,b)}|\phi\rangle\right)|AUX\rangle$ , for some ancillary state  $|AUX\rangle \in \mathcal{H}'$ .

This definition is rather informal because whenever the number of possible questions and answers is fixed and independent of the security parameter (as is the case in this paper), single-round question-answer correlations  $p(a, b|x, y)$  alone cannot be sufficient: a prover can always succeed in the protocol simply by answering the verifier's questions according to a look-up table; such a prover is classical and does not actually perform any computation. Therefore, our protocol will have multiple rounds of interaction between the verifier and the prover: the questions and answers in the initial “setup rounds” will involve a public key that scales with the security parameter; then, in the last round, the verifier observes question-answer correlations  $p(a, b|x, y)$  similar to standard self-testing, i.e., with a fixed question and answer length. Instead of using multi-round interaction, one could also try to build a single-round protocol with questions that depend on the security parameter (e.g., the question would include a public key). A number of recent works have shown that under the (quantum) random oracle assumption, the protocol for certifying the quantumness of a prover from [8] and the verification protocol from [23] can be adapted to this single-round setting [2, 11, 9]. We leave it for future work to investigate whether the interaction in our protocol can also be removed with the random oracle assumption.

To obtain a statement that is more similar to the two-prover scenario, we consider the stronger constraint that the desired measurements have a tensor product form  $K_{x,y}^{(a,b)} = M_x^{(a)} \otimes N_y^{(b)}$ . In particular, this means that answer  $a$  only depends on question  $x$  and  $b$  only depends on  $y$ , and it enforces a natural tensor product structure on the prover's space. Specifically, we define Hilbert spaces  $\mathcal{H}_A, \mathcal{H}_B$  and  $\mathcal{H}'$  and deduce the existence of an isometry  $V$  from the prover's physical space  $\mathcal{H}$  to  $\mathcal{H}_A \otimes \mathcal{H}_B \otimes \mathcal{H}'$  such that under the isometry, the measurements operators  $P_{x,y}^{(a,b)}$  act on  $|\psi\rangle$  in the same way that tensor product measurement operators of the form  $M_x^{(a)} \otimes N_y^{(b)}$  act on  $|\phi\rangle_{AB}$ , where  $M_x^{(a)}$  acts only on  $\mathcal{H}_A$ ,  $N_y^{(b)}$  acts only on  $\mathcal{H}_B$ , and  $|\phi\rangle_{AB}$  is the state that we are self-testing for (e.g., a Bell state).

► **Definition 4** (Self-testing of tensor product strategies for a single prover, informal). *The correlations  $p(a, b|x, y)$  self-test a state  $|\phi\rangle_{AB}$  and measurements  $\{M_x^{(a)}\}$  on system  $A$  and  $\{N_y^{(b)}\}$  on system  $B$  if for any state  $|\psi\rangle \in \mathcal{H}$  and measurements  $\{P_{x,y}^{(a,b)}\}_{a,b}$  on  $\mathcal{H}$  that realise the correlations  $p(a, b|x, y)$ , there exists an isometry  $V : \mathcal{H} \rightarrow \mathcal{H}_A \otimes \mathcal{H}_B \otimes \mathcal{H}'$  such that*

1.  $V|\psi\rangle = |\phi\rangle_{AB}|AUX\rangle$ ,
2.  $VP_{x,y}^{(a,b)}|\psi\rangle = \left((M_x^{(a)} \otimes N_y^{(b)})|\phi\rangle_{AB}\right)|AUX\rangle$ , for some ancillary state  $|AUX\rangle \in \mathcal{H}'$ .

Again, this definition is informal for the same reason as for Definition 3. A formal statement of such a single-prover self-testing result with a tensor product structure is given in [27, Theorem 4.38], the main result of our work.

## 1.2 Cryptographic primitives

The main cryptographic primitive underlying our self-testing protocol is a so-called extended noisy trapdoor claw-free function family (ENTCF family). ENTCF families were introduced by Mahadev in [23], building on the construction of noisy trapdoor claw-free function families by Brakerski et al. in [8]. Here, we only give a brief informal description of the main properties of an ENTCF family (see [27, Section 2.2] for references and details).

An ENTCF family consists of two families  $\mathcal{F}$  and  $\mathcal{G}$  of function pairs. A function pair  $(f_{k,0}, f_{k,1}) \in \mathcal{F}$  is called a *claw-free pair* and is indexed by a public key  $k$ . Similarly, an *injective pair* is a pair of functions  $(f_{k,0}, f_{k,1}) \in \mathcal{G}$ , also indexed by a public key  $k$ . Informally, the most important properties are the following:

1. For fixed  $k \in \mathcal{K}_{\mathcal{F}}$ ,  $f_{k,0}$  and  $f_{k,1}$  are bijections with the same image, i.e., for every  $y$  in their image there exists a unique pair  $(x_0, x_1)$ , called a *claw*, such that  $f_{k,0}(x_0) = f_{k,1}(x_1) = y$ .
2. Given a key  $k \in \mathcal{K}_{\mathcal{F}}$  for a claw-free pair, it is quantum-computationally intractable (without access to trapdoor information) to compute both a preimage  $x_i$  and a single generalised bit of  $x_0 \oplus x_1$  (i.e.,  $d \cdot (x_0 \oplus x_1)$  for any non-trivial bit string  $d$ ), where  $(x_0, x_1)$  forms a valid claw. This is called the *adaptive hardcore bit property*.
3. For fixed  $k \in \mathcal{K}_{\mathcal{G}}$ ,  $f_{k,0}$  and  $f_{k,1}$  are injective functions with disjoint images.
4. Given a key  $k \in \mathcal{K}_{\mathcal{F}} \cup \mathcal{K}_{\mathcal{G}}$ , it is quantum-computationally hard (without access to trapdoor information) to determine the “function type”, i.e., to decide whether  $k$  is a key for a claw-free or an injective pair. This is called *injective invariance*.
5. For every key  $k \in \mathcal{K}_{\mathcal{F}} \cup \mathcal{K}_{\mathcal{G}}$ , there exists a trapdoor  $t_k$ , which can be sampled together with  $k$  and with which (ii) and (iv) are computationally easy.

## 2 Our self-testing protocol

We now give an informal description of our self-testing protocol with the honest prover behaviour and provide some intuition for its soundness. A full description of the protocol is given in [27, Figure 1].

On a very high level, one can view the protocol as first executing the RSP protocol from [19] twice in parallel to prepare two qubits in the provers space. Then, the prover is asked to perform an entangling operation on these two qubits. Because the prover does not know which states the qubits are in, and the entangling operation acts differently on different states, to pass the checks in the protocol the prover has to apply the entangling operation honestly.

In more detail, the protocol begins with the verifier sampling two uniformly random bits  $\theta_1, \theta_2$ , each bit denoting a *basis choice* (either the computational or the Hadamard basis). The case where both bits denote the Hadamard basis will be the one where the prover prepares a Bell pair, whereas the other basis choices serve as tests that prevent the prover from cheating. Depending on these basis choices, the verifier then samples two key-trapdoor pairs  $(k_1, t_{k_1})$  and  $(k_2, t_{k_2})$  from the ENTCF family: for the computational basis, it samples an injective pair, and for the Hadamard basis a claw-free pair. The verifier sends the keys to the prover and keeps the trapdoors private.

The honest prover treats the two keys separately. For each key  $k_i$ , he prepares the state

$$|\psi_i\rangle = \frac{1}{\sqrt{2|\mathcal{X}|}} \sum_{x \in \mathcal{X}, b \in \{0,1\}} |b\rangle |x\rangle |f_{k_i,b}(x)\rangle. \quad (1)$$

Here,  $\mathcal{X}$  is the domain of the ENTCF family. Note that even though the prover does not know which kind of function (claw-free or injective) he is dealing with, the definition of ENTCF families still allows him to construct this state. The prover now measures both

image registers (i.e., the registers storing “ $f_{k_i,b}(x)$ ”), obtains images  $y_1, y_2$ , and sends these to the verifier. (In the terminology of [23], this is called a “commitment”.) Depending on the choice of function family by the verifier, the prover’s post-measurement state has one of two forms: if the verifier sampled the key  $k_i$  from the injective family, the post-measurement state is a computational basis state:

$$|\psi'_i\rangle = |b\rangle|x_b\rangle, \quad (2)$$

where  $x_b$  is the unique preimage of  $y_i$ . If the key  $k_i$  belongs to a claw-free family, the post-measurement state is a superposition over a claw:

$$|\psi'_i\rangle = \frac{1}{\sqrt{2}}(|0\rangle|x_0\rangle + |1\rangle|x_1\rangle), \quad (3)$$

where  $(x_0, x_1)$  form a claw, i.e.,  $f_{k,0}(x_0) = f_{k,1}(x_1) = y$ .

At this point, the verifier selects a round type, either a “preimage round” or a “Hadamard round”, uniformly at random and sends the round type to the prover. For a preimage round, the honest prover measures his entire state in the computational basis and returns the result; the verifier checks that the prover has indeed returned correct preimages for the submitted  $y_1, y_2$ . The preimage round is an additional test that is required for us to leverage the adaptive hardcore bit property, but we do not discuss this further in this overview.

For a Hadamard round, the honest prover measures both of his preimage registers (i.e., the registers containing “ $x_b$ ”) in the Hadamard basis, obtains two bit strings  $d_1, d_2$ , and sends these to the verifier. This results in the following states (using the notation from above):

$$|\psi''_i\rangle = \begin{cases} |b\rangle & \text{if } k_i \text{ belongs to an injective family,} \\ \frac{1}{\sqrt{2}}(|0\rangle + (-1)^{d_i \cdot (x_0 \oplus x_1)}|1\rangle) & \text{if } k_i \text{ belongs to a claw-free family.} \end{cases} \quad (4)$$

Note that the phase in the second case is exactly the adaptive hardcore bit from the definition of ENTFC families. At this point, the verifier selects two additional bases  $q_1, q_2$  uniformly at random (again from either the computational or Hadamard basis), and sends these to the prover. In analogy to self-testing, we call these bases “questions”. The honest prover now applies a  $CZ$  gate (an entangling two-qubit gate that applies a  $\sigma_Z$  operation to the second qubit if the first qubit is in state  $|1\rangle$ ) to its state  $|\psi''_1\rangle|\psi''_2\rangle$ . In the case where both  $\theta_1$  and  $\theta_2$  specify the Hadamard basis, this results in a Bell state (rotated by a single-qubit Hadamard gate). The prover measures the individual qubits of the resulting state in the bases specified by  $q_1, q_2$ . The outcomes  $v_1, v_2$  are returned to the verifier.

The verifier can use the prover’s answers  $y_1, y_2, d_1, d_2$  and her trapdoor information  $t_{k_1}, t_{k_2}$  to determine which state  $CZ|\psi''_1\rangle|\psi''_2\rangle$  the prover should have prepared. The verifier accepts the prover if his answers  $v_1, v_2$  are consistent with making the measurements specified by  $q_1, q_2$  on the honest prover’s state  $CZ|\psi''_1\rangle|\psi''_2\rangle$ .

## 2.1 Soundness proof

We now give a brief intuition for the soundness of the protocol; the full soundness proof is given in [27, Section 4]. Let us first consider a version of the protocol where the prover is not supposed to perform a  $CZ$  operation. As noted before, this would be (a simplified version of) the RSP protocol [19], executed twice in parallel. For the purposes of this overview, let us assume that the only way for the prover to pass these two parallel executions of the RSP protocol is to treat each execution separately, i.e., use a tensor product Hilbert space  $\mathcal{H}_1 \otimes \mathcal{H}_2$  and execute each instance of the RSP protocol on a different part of the space

(enforcing such a tensor product structure is reminiscent of the classic question of parallel repetition [30] and is actually one of the main difficulties in our soundness proof, but we leave the details of this for [27, Section 4]). It now follows from the security of the RSP protocol that the prover must have prepared one of  $\{|0\rangle, |1\rangle, |+\rangle, |-\rangle\}$  in each part of his space (up to a “local” change of basis for each space), but he does not know which one.

Now consider how a  $CZ$  operation acts on these different states: if both states are Hadamard basis states (e.g.,  $|+\rangle|-\rangle$ ), the  $CZ$  operation will entangle them and produce a Bell state (rotated by a single-qubit Hadamard gate); in contrast, if at least one of the states is a computational basis state (e.g.,  $|1\rangle|-\rangle$ ), the resulting state will still be a product state of computational and Hadamard basis states (albeit a different one). This means that in the latter case, the  $CZ$  operation essentially only relabels the states. Therefore, if the verifier adapts her checks to account for the relabelling, in the latter case the guarantees from the RSP protocol still hold. Because the prover does not know which bases the verifier has selected, we can extend these guarantees to the case of two Hadamard basis states, too.

We stress that this only provides a rough intuition, and that the actual proof proceeds quite differently from this because we cannot just assume the existence of a tensor product structure on the prover’s Hilbert space. Deducing this tensor product structure poses technical difficulties. In two-prover self-testing proofs, the first step is to show that the measurement operators used by each prover approximately satisfy certain relations, e.g. anti-commutation. Because the measurement operators of different provers act on different Hilbert spaces, they exactly commute. Combining the approximate relations from the first step with the exact commutation relations, one can show that the prover’s measurement operators must be close to some desired operators, e.g. the Pauli operators. This last “rounding step” typically uses Jordan’s lemma or a stability theorem for approximate group representations [20]. In our case, we cannot show exact commutation relations between operators – commutation can only be enforced via the protocol, which tolerates a small failure probability. Hence, we are only able to show approximate commutation relations, which prevents us from applying Jordan’s lemma or the result of [20]. We therefore develop an alternative approach to “rounding” the prover’s operators that only requires approximate commutation and leverages the cryptographic assumptions. This method might also be useful for other applications that require a very tight “cryptographic leash” on a quantum prover.

### 3 Discussion

Self-testing has developed into a versatile tool for quantum information processing and quantum complexity theory and presents one of the strongest possible black-box certification techniques of quantum devices. The standard self-testing setting involves multiple non-communicating quantum provers, which is difficult to enforce in practice. The main contribution of this paper is the construction of a self-testing protocol that allows a classical verifier to certify that a single computationally bounded quantum prover has prepared a Bell state and measured the individual qubits of the state in the computational or Hadamard basis, up to a global change of basis applied to both the state and measurements. This means that we are able to certify the existence of entanglement in a single quantum device.<sup>4</sup>

<sup>4</sup> The freedom of applying a global change of basis means that the entangled Bell state can be mapped to a product state. However, then the prover’s measurements are mapped to entangling measurements, so entanglement is still present.

Due to the interactive nature of our protocol, this certification remains valid even if it turned out that any quantum computation is classically simulable, i.e.,  $\text{BQP} = \text{BPP}$ .<sup>5</sup> It therefore constitutes a “test of quantumness” in the sense of [8] and differs from proposals for testing quantum supremacy such as [7], which only certify the “quantumness” of a device under the assumption that  $\text{BQP} \neq \text{BPP}$ .<sup>6</sup>

Existing multi-prover self-testing protocols are typically based on non-local games, e.g., the CHSH game [25]. Our self-testing protocol follows a more “custom” approach guided by the available cryptographic primitives. While this enables us to construct a single-prover self-test for single-qubit measurements on a Bell state, arguably the most important quantum state for many applications, it does not allow us to extend the result to other states for which multi-prover self-tests are known [13]. To better make use of the extensive existing self-testing literature, it would be desirable to construct a procedure that allows for the “translation” of multi-prover non-local games to single-prover games with computational assumptions. In classical cryptography, similar attempts have been made to construct single-prover argument systems from multi-prover proof systems using fully homomorphic encryption [1, 22, 18].

Another approach to constructing single-prover self-tests for a larger class of states might be to strengthen Mahadev’s measurement protocol [23] from guaranteeing the existence of a state compatible with the measurement results to certifying that the prover actually has prepared this state. As a step in this direction, the second author and Zhang recently showed that Mahadev’s protocol is a classical proof of quantum knowledge [37]. The concept of a proof of quantum knowledge, first introduced in [10, 15] for the setting of a quantum verifier and extended to the setting of a classical verifier in [37], is still less stringent than a self-test and in particular lacks the strong characterisation of the prover’s measurements that we obtain in self-testing.

Beyond the conceptual appeal of gaining more fine-grained control over untrusted quantum devices, our self-testing protocol presents a first step towards translating multi-prover protocols for applications such as delegated computation [32, 14], randomness expansion [16, 36, 28], or secure multi-party quantum computation [17, 4] to a single-prover setting. There are already computationally secure single-prover protocols for delegated quantum computation [23] and randomness expansion [8]; however, establishing a more general link between self-testing-based multi-prover protocols and computationally secure single-prover protocols is still desirable: it might lead to conceptually simpler single-prover protocols and will be useful for constructing single-prover protocols for other applications without resorting to a low-level cryptographic analysis.

For example, using our self-testing theorem in a black-box way, the first author and others have recently constructed a protocol for device-independent quantum key distribution (DIQKD) [26]. In contrast to previous DIQKD protocols, which rely on a non-communication similar to the one in standard self-testing, this new DIQKD protocol requires no non-communication assumption and more closely models how DIQKD protocols are expected to be implemented experimentally. Crucially, the security analysis of this DIQKD protocol can be reduced to our self-testing theorem without any intricate cryptographic analysis involving computational hardness assumptions.

<sup>5</sup> Note that the LWE assumption is independent of whether  $\text{BQP} = \text{BPP}$  or not, since LWE is assumed to be hard for both quantum and classical computers.

<sup>6</sup> Intuitively, the reason for this is the following: in our protocol and in [8], the quantum prover has to be able to compute either a preimage or a pair  $(u, d)$  such that  $u = d \cdot (x_0 \oplus x_1)$ , where  $(x_0, x_1)$  forms a claw. If a classical prover was able to correctly compute a preimage or a pair  $(u, d)$ , it could be rewound to compute both at the same time, contradicting the adaptive hardcore bit property. In a quantum prover, the collapsing nature of quantum measurements prevents us from rewinding the prover.



We believe that, in a similar vein, our protocol will also serve as a useful building block for other future protocols for computationally bounded quantum devices, in the same way that self-testing for EPR pairs in the multi-prover scenario has proved to be a versatile tool in physics, cryptography, and complexity theory.

## References

- 1 William Aiello, Sandeep Bhatt, Rafail Ostrovsky, and S. Raj. Rajagopalan. Fast Verification of Any Remote Procedure Call: Short Witness-Indistinguishable One-Round Proofs for NP. *Automata, Languages and Programming - ICALP 2000, Lecture Notes in Computer Science, Springer*, pages 463–474, 2000. doi:10.1007/3-540-45022-X\_39.
- 2 Gorjan Alagic, Andrew M Childs, Alex B Grilo, and Shih-Han Hung. Non-interactive classical verification of quantum computation. *arXiv preprint*, 2019. arXiv:1911.08101.
- 3 John S. Bell. On the Einstein Podolsky Rosen paradox. *Physics Physique Fizika*, 1(3):195–200, November 1964. doi:10.1103/PhysicsPhysiqueFizika.1.195.
- 4 Michael Ben-Or, Claude Crépeau, Daniel Gottesman, Avinatan Hassidim, and Adam Smith. Secure multiparty quantum computation with (only) a strict honest majority. *IEEE 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 249–260, 2006. doi:10.1109/FOCS.2006.68.
- 5 Kishor Bharti, Maharshi Ray, Antonios Varvitsiotis, Adán Cabello, and Leong-Chuan Kwek. Local certification of programmable quantum devices of arbitrary high dimensionality. *arXiv preprint*, 2019. arXiv:1911.09448.
- 6 Kishor Bharti, Maharshi Ray, Antonios Varvitsiotis, Naqeeb Ahmad Warsi, Adán Cabello, and Leong-Chuan Kwek. Robust self-testing of quantum systems via noncontextuality inequalities. *Phys. Rev. Lett.*, 122:250403, June 2019. doi:10.1103/PhysRevLett.122.250403.
- 7 Adam Boulund, Bill Fefferman, Chinmay Nirkhe, and Umesh Vazirani. On the complexity and verification of quantum random circuit sampling. *Nature Physics*, 15(2):159–163, February 2019. doi:10.1038/s41567-018-0318-2.
- 8 Z. Brakerski, P. Christiano, U. Mahadev, U. Vazirani, and T. Vidick. A cryptographic test of quantumness and certifiable randomness from a single quantum device. *IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 320–331, 2018. doi:10.1109/FOCS.2018.00038.
- 9 Zvika Brakerski, Venkata Koppula, Umesh Vazirani, and Thomas Vidick. Simpler proofs of quantumness. *arXiv preprint*, 2020. arXiv:2005.04826.
- 10 Anne Broadbent and Alex B Grilo. Zero-knowledge for QMA from locally simulatable proofs. *arXiv preprint*, 2019. arXiv:1911.07782.
- 11 Nai-Hui Chia, Kai-Min Chung, and Takashi Yamakawa. Classical verification of quantum computations with efficient verifier. *arXiv preprint*, 2019. arXiv:1912.00990.
- 12 Alexandru Cojocaru, Léo Colisson, Elham Kashefi, and Petros Wallden. QFactory: Classically-Instructed Remote Secret Qubits Preparation. *Advances in Cryptology - ASIACRYPT 2019, Lecture Notes in Computer Science, Springer*, pages 615–645, 2019. doi:10.1007/978-3-030-34578-5\_22.
- 13 Andrea Coladangelo, Koon Tong Goh, and Valerio Scarani. All pure bipartite entangled states can be self-tested. *Nature Communications*, 8(1):15485, August 2017. doi:10.1038/ncomms15485.
- 14 Andrea Coladangelo, Alex B. Grilo, Stacey Jeffery, and Thomas Vidick. Verifier-on-a-leash: New schemes for verifiable delegated quantum computation, with quasilinear resources. *Advances in Cryptology - EUROCRYPT 2019, Lecture Notes in Computer Science, Springer*, 11478 LNCS:247–277, 2019. doi:10.1007/978-3-030-17659-4\_9.
- 15 Andrea Coladangelo, Thomas Vidick, and Tina Zhang. Non-interactive zero-knowledge arguments for QMA, with preprocessing. In *Annual International Cryptology Conference*, pages 799–828. Springer, 2020.
- 16 Roger Colbeck. *Quantum and relativistic protocols for secure multi-party computation*. PhD Thesis, University of Cambridge, 2006. arXiv:0911.3814.

- 17 Claude Crépeau, Daniel Gottesman, and Adam Smith. Secure multi-party quantum computation. *Proceedings of the 34th Annual ACM Symposium on Theory of Computing*, pages 643–652, 2002. doi:10.1145/509907.510000.
- 18 Yevgeniy Dodis, Shai Halevi, Ron D. Rothblum, and Daniel Wichs. Spooky Encryption and Its Applications. *Advances in Cryptology - CRYPTO 2016, Lecture Notes in Computer Science, Springer*, pages 93–122, 2016. doi:10.1007/978-3-662-53015-3\_4.
- 19 Alexandru Gheorghiu and Thomas Vidick. Computationally-secure and composable remote state preparation. In *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1024–1033. IEEE, 2019.
- 20 William T. Gowers and Omid Hatami. Inverse and stability theorems for approximate representations of finite groups. *Sbornik: Mathematics*, 208(12):1784, 2017. doi:10.1070/SM8872.
- 21 Zhengfeng Ji, Anand Natarajan, Thomas Vidick, John Wright, and Henry Yuen.  $MIP^* = RE$ . *arXiv preprint*, 2020. arXiv:2001.04383.
- 22 Yael Tauman Kalai, Ran Raz, and Ron D. Rothblum. How to delegate computations: The power of no-signaling proofs. *Proceedings of the 46th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 485–494, 2014. doi:10.1145/2591796.2591809.
- 23 Urmila Mahadev. Classical verification of quantum computations. *IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 259–267, 2018. doi:10.1109/FOCS.2018.00033.
- 24 Dominic Mayers and Andrew Yao. Self testing quantum apparatus. *Quantum Info. Comput.*, 4(4):273–286, July 2004. URL: <https://dl.acm.org/doi/10.5555/2011827.2011830>.
- 25 M McKague, T H Yang, and V Scarani. Robust self-testing of the singlet. *Journal of Physics A: Mathematical and Theoretical*, 45(45):455304, October 2012. doi:10.1088/1751-8113/45/45/455304.
- 26 Tony Metger, Yfke Dulek, Andrea Coladangelo, and Rotem Arnon-Friedman. Device-independent quantum key distribution from computational assumptions. *arXiv preprint*, 2020. arXiv:2010.04175.
- 27 Tony Metger and Thomas Vidick. Self-testing of a single quantum device under computational assumptions. *CoRR*, 2020. (Full version: arXiv:2001.09161).
- 28 Carl A. Miller and Yaoyun. Shi. Universal security for randomness expansion from the spot-checking protocol. *SIAM Journal on Computing*, 46(4):1304–1335, 2017. doi:10.1137/15M1044333.
- 29 Sandu Popescu and Daniel Rohrlich. Which states violate Bell’s inequality maximally? *Physics Letters A*, 169(6):411–414, October 1992. doi:10.1016/0375-9601(92)90819-8.
- 30 Ran Raz. A parallel repetition theorem. *SIAM Journal on Computing*, 27(3):763–803, 1998. doi:10.1137/S0097539795280895.
- 31 Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6), 2009. doi:10.1145/1568318.1568324.
- 32 Ben W Reichardt, Falk Unger, and Umesh Vazirani. Classical command of quantum systems. *Nature*, 496(7446):456, 2013. doi:10.1038/nature12035.
- 33 Valerio Scarani. *Bell Nonlocality*. Oxford University Press, 2019.
- 34 Stephen J. Summers and Reinhard Werner. Maximal violation of Bell’s inequalities is generic in quantum field theory. *Communications in Mathematical Physics*, 110(2):247–259, June 1987. doi:10.1007/BF01207366.
- 35 Ivan Šupić and Joseph Bowles. Self-testing of quantum systems: a review. *Quantum*, 4:337, 2020.
- 36 Umesh Vazirani and Thomas Vidick. Certifiable quantum dice: Or, true random number generation secure against quantum adversaries. *Proceedings of the 44th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 61–76, 2012. doi:10.1145/2213977.2213984.
- 37 Thomas Vidick and Tina Zhang. Classical proofs of quantum knowledge. *arXiv preprint*, 2020. arXiv:2005.01691.

# Polynomial-Time Trace Reconstruction in the Low Deletion Rate Regime

**Xi Chen**

Columbia University, New York, NY, USA  
<http://www.cs.columbia.edu/~xichen>  
xichen@cs.columbia.edu

**Anindya De**

University of Pennsylvania, Philadelphia, PA, USA  
<https://www.seas.upenn.edu/~anindyad/>  
anindyad@cis.upenn.edu

**Chin Ho Lee**

Columbia University, New York, NY, USA  
<https://www.cs.columbia.edu/~chlee/>  
c.h.lee@columbia.edu

**Rocco A. Servedio**

Columbia University, New York, NY, USA  
<http://www.cs.columbia.edu/~rocco>  
rocco@cs.columbia.edu

**Sandip Sinha** 

Columbia University, New York, NY, USA  
<https://sites.google.com/view/sandips>  
sandip@cs.columbia.edu

---

## Abstract

In the *trace reconstruction problem*, an unknown source string  $x \in \{0, 1\}^n$  is transmitted through a probabilistic *deletion channel* which independently deletes each bit with some fixed probability  $\delta$  and concatenates the surviving bits, resulting in a *trace* of  $x$ . The problem is to reconstruct  $x$  given access to independent traces. Trace reconstruction of arbitrary (worst-case) strings is a challenging problem, with the current state of the art for  $\text{poly}(n)$ -time algorithms being the 2004 algorithm of Batu et al. [2]. This algorithm can reconstruct an arbitrary source string  $x \in \{0, 1\}^n$  in  $\text{poly}(n)$  time provided that the deletion rate  $\delta$  satisfies  $\delta \leq n^{-(1/2+\varepsilon)}$  for some  $\varepsilon > 0$ .

In this work we improve on the result of [2] by giving a  $\text{poly}(n)$ -time algorithm for trace reconstruction for any deletion rate  $\delta \leq n^{-(1/3+\varepsilon)}$ . Our algorithm works by alternating an alignment-based procedure, which we show effectively reconstructs portions of the source string that are not “highly repetitive”, with a novel procedure that efficiently determines the length of highly repetitive subwords of the source string.

**2012 ACM Subject Classification** Mathematics of computing → Probabilistic inference problems

**Keywords and phrases** trace reconstruction

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.20

**Related Version** Full Version: <https://arxiv.org/abs/2012.02844>

**Funding** *Xi Chen*: Supported by NSF grants CCF-1703925 and IIS-1838154.

*Anindya De*: Supported by NSF grants CCF-1926872 and CCF-1910534.

*Chin Ho Lee*: Supported by a grant from the Croucher Foundation and by the Simons Collaboration on Algorithms and Geometry.

*Rocco A. Servedio*: Supported by NSF grants CCF-1814873, IIS-1838154, CCF-1563155, and by the Simons Collaboration on Algorithms and Geometry.

*Sandip Sinha*: Supported by NSF grants CCF-1714818, CCF-1822809, IIS-1838154, CCF-1617955, CCF-1740833, and by the Simons Collaboration on Algorithms and Geometry.



© Xi Chen, Anindya De, Chin Ho Lee, Rocco A. Servedio, and Sandip Sinha;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 20; pp. 20:1–20:20



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

## 1 Introduction

The *trace reconstruction* problem was proposed almost twenty years ago in works of [10, 11, 2], though some earlier variants of the problem were already considered in the 1970s [9]. This problem deals with the *deletion channel*, which works as follows: when an  $n$ -bit string (the source string) is passed through a deletion channel of rate  $\delta$ , each coordinate is independently deleted with probability  $\delta$ . The surviving  $n' \leq n$  coordinates are concatenated to form the output of the channel, which is referred to as a *trace* of the original source string; we write “ $\mathbf{z} \sim \text{Del}_\delta(x)$ ” to indicate that  $\mathbf{z}$  is a trace generated from source string  $x$  according to this probabilistic process. As discussed in [13], this channel provides an elegant formalization for the theoretical study of problems involving synchronization errors.

In the *trace reconstruction problem*, independent traces are generated from an unknown and arbitrary source string  $x \in \{0, 1\}^n$ , and the task of the algorithm is to reconstruct (with high probability)  $x$  from its traces. The trace reconstruction problem is motivated by applications in several domains, including sensor networks and biology [13, 1, 16, 15]. It is also attractive because it is a clean and natural “first problem” which already seems to capture much of the difficulty of dealing with the deletion channel.

The problem of trace reconstruction for an arbitrary (worst-case) source string  $x$  has proved to be quite challenging.<sup>1</sup> [2] gave an algorithm that runs in  $\text{poly}(n)$  time, uses  $\text{poly}(n)$  traces, and with high probability reconstructs an arbitrary source string  $x \in \{0, 1\}^n$  provided that the deletion rate  $\delta$  is at most  $n^{-(1/2+\varepsilon)}$  for some constant  $\varepsilon > 0$ . Unfortunately, the trace reconstruction problem seems to quickly become intractable at higher deletion rates. Holenstein et al. [8] gave an algorithm that runs in time  $\exp(O(n^{1/2}))$  and uses  $\exp(O(n^{1/2}))$  traces for any deletion rate  $\delta$  that is bounded away from 1 by a constant, and this result was subsequently improved in simultaneous and independent works by [4, 14], both of which gave algorithms with time and sample complexity  $\exp(O(n^{1/3}))$ . On the lower bounds side, for  $\delta = \Theta(1)$  successively stronger lower bounds on the required sample complexity were given by [12] and [5], with the current state of the art being a  $\tilde{\Omega}(n^{3/2})$  lower bound due to Chase [3].

**The low deletion rate regime.** The positive result of [4] actually gives an algorithm that is faster than  $\exp(O(n^{1/3}))$  if the deletion rate is sufficiently low: [4] shows that for  $O(\log^3 n)/n \leq \delta \leq 1/2$ , their algorithm runs in time  $\exp(O(\delta n)^{1/3})$ . Consequently, for the specific deletion rate  $\delta = n^{-(1/2+\varepsilon)}$ , the [4] algorithm runs in time essentially  $\exp(O(n^{1/6}))$ , and [4] shows that no faster running time or better sample complexity is possible for any “mean-based” algorithm, a class of algorithms which includes those of [4, 14, 8].

Algorithmic approaches other than mean-based algorithms can provably do better at low deletion rates. This is witnessed by the algorithm of Batu et al. [2] which, as described earlier, runs in  $\text{poly}(n)$  time and uses  $\text{poly}(n)$  samples at deletion rate  $\delta = n^{-(1/2+\varepsilon)}$ . The main algorithmic component of [2] is a “*Bitwise Majority Alignment*” (BMA for short) procedure, which is further augmented with a simple procedure to determine the length of long “runs” (subwords of  $x$  of the form  $0^\ell$  or  $1^\ell$  with  $\ell \geq \sqrt{n}$ ). Roughly speaking, the BMA algorithm maintains a pointer in each trace and increments those pointers in successive time steps, attempting to always keep almost all of the pointers correctly aligned together. The analysis of [2] shows that the BMA algorithm succeeds if the source string  $x$  does not contain any

---

<sup>1</sup> We note that the average-case problem, in which the reconstruction algorithm is only required to succeed for a  $1 - o(1)$  fraction of all possible source strings in  $\{0, 1\}^n$ , is much more tractable, with the current state of the art [6, 7] being an algorithm that uses  $\exp(O(\log^{1/3} n))$  traces and runs in  $\text{poly}(n)$  time for any deletion rate  $\delta$  that is bounded away from 1.

long runs, but a challenge for the BMA algorithm is that the pointers in different traces will inevitably become misaligned if  $x$  does contain a long run  $0^\ell$  or  $1^\ell$ ; this is why the [2] algorithm must interleave BMA with a procedure to handle long runs separately. Intuitively, deletion rate  $\delta = n^{-1/2}$  is a barrier for the [2] analysis because if  $\delta = \omega(n^{-1/2})$ , then each trace is likely to have multiple locations where more than one consecutive bit of  $x$  is “dropped,” which is problematic for the analysis of BMA given in [2].

To summarize: given this state of the art from prior work, it is clear that alignment-based approaches can outperform mean-based algorithms at low deletion rates, but it is not clear whether, or how far, alignment-based approaches can be extended beyond the [2] results. Further incentive for studying the low deletion rate regime comes from potential applications in areas such as computer networks, where it may be natural to model deletions as occurring at relatively low rates. These considerations motivate the results of the present paper, which we now describe.

### 1.1 This work: An improved algorithm for the low deletion rate regime

The main result of this paper is an efficient algorithm that can handle significantly higher deletion rates than the [2] algorithm. We prove the following:

► **Theorem 1** (Efficient trace reconstruction at deletion rate  $\delta \geq n^{-(1/3+\varepsilon)}$ ). *Fix any constant  $\varepsilon > 0$  and let  $\delta = n^{-(1/3+\varepsilon)}$ . There is an algorithm **Reconstruct** that uses  $O(n^{4/3})$  independent traces drawn from  $\text{Del}_\delta(x)$  (where  $x \in \{0,1\}^n$  is arbitrary and unknown to **Reconstruct**), runs in  $O(n^{7/3})$  time, and outputs the unknown source string  $x$  with probability at least  $9/10$ .*

Note that any deletion rate  $\delta < n^{-(1/3+\varepsilon)}$  can of course be handled, given Theorem 1, by simply deleting additional bits to reduce to the  $\delta = n^{-(1/3+\varepsilon)}$  case. Note further that any desired success probability  $1 - \kappa$  can easily be achieved from Theorem 1 by running **Reconstruct**  $O(\log(1/\kappa))$  times and then taking a majority vote.

At a high level, the **Reconstruct** algorithm works by interleaving two different subroutines.

- The first subroutine is (essentially) the BMA algorithm, for which we provide an improved analysis, showing that BMA successfully reconstructs any string that does not contain a long subword (of length at least  $M = 2m + 1$  with  $m = n^{1/3}$ ) that is a prefix of  $s^\infty$  for some short (constant-length) bitstring  $s$ . We refer to long and “highly-repetitive” subwords of  $x$  of this form as “ $s$ -deserts” of  $x$ ; see Definition 2 for a detailed definition.
- The second subroutine is a new algorithm which we show efficiently determines the length of an  $s$ -desert in the source string  $x$ .

Thus, two novel aspects of this work that go beyond [2] are (i) our improved analysis of BMA, and (ii) our new procedure for efficiently measuring deserts (the analogous component of the [2] algorithm could only measure runs, which correspond to  $s$ -deserts with  $|s| = 1$ ).

We believe that it may be possible to further extend the kind of “hybrid” approach that we employ in this paper to obtain efficient trace reconstruction algorithms that can handle even larger deletion rates  $\delta$ . However, there are some significant technical challenges that would need to be overcome in order to do so. We describe some of these challenges at the end of the next section, which gives a more detailed overview of our approach.

## 2 Overview of our approach

As alluded to in the introduction, at a high level our algorithm carries out a careful interleaving of two procedures, which we call **BMA** and **FindEnd**. In this section we first give a high-level overview of the procedure BMA as well as our improved analysis. Then we give a high-level overview of **FindEnd**, and finally we explain how these two procedures are interleaved. We close with a brief discussion of possibilities and barriers to further progress.

## 2.1 Overview of BMA

The procedure **BMA** is exactly the same as the bitwise majority alignment algorithm of [2]; our new contribution regarding **BMA** is in giving a more general analysis. To explain the high level idea, let us fix the deletion rate  $\delta = n^{-(1/3+\varepsilon)}$  and a constant  $C = \lceil 100/\varepsilon \rceil$ . Let

$$m = n^{1/3} \quad \text{and} \quad M = 2m + 1.$$

The **BMA** procedure operates on a sample of some  $N = O(\log n)$  traces  $\mathbf{y}^1, \dots, \mathbf{y}^N$  drawn independently from  $\text{Del}_\delta(x)$  before the procedure begins its execution. Note that for any  $i \in [N]$  and any position  $\ell$  in the trace  $\mathbf{y}^i$ , there is a position  $f_i(\ell)$  satisfying  $\ell \leq f_i(\ell) \leq n-1$  in the target string  $x = (x_0, \dots, x_{n-1})$  that maps to  $\ell$  under the deletion process.<sup>2</sup> The high level idea of **BMA** is to maintain pointers  $\text{current}_1, \dots, \text{current}_N$ , with  $\text{current}_i$  pointing to a position in  $\mathbf{y}^i$ , such that *most of them are correctly aligned* – i.e., at the beginning of each time step  $t$ ,  $t = 0, 1, \dots$ , as we try to determine  $x_t$ , we have  $f_i(\text{current}_i) = t$  for most  $i \in [N]$ . Note that if this alignment guarantee were to hold for more than half of the traces for all  $t = 0, 1, \dots, n-1$ , then we could reconstruct the unknown string  $x$  by taking a majority vote of  $\mathbf{y}_{\text{current}_i}^i$  in each time step. Indeed we show that this happens with high probability over the randomness of  $\mathbf{y}^1, \dots, \mathbf{y}^N$  when  $x$  does not contain an  $s$ -desert for any string  $s \in \{0, 1\}^{\leq C}$  (i.e. a subword of length at least  $M = 2m + 1$  that is a prefix of  $s^\infty$ ). In contrast, the analysis of [2] requires the deletion rate to be  $n^{-(1/2+\varepsilon)}$  but works as long as  $x$  does not contain a run of 0's or 1's (or  $s$ -deserts with  $|s| = 1$  in our notation) of length at least  $\sqrt{n}$ .

To explain **BMA** in more detail, let us initialize  $t = 0$  and pointers  $\text{current}_1(0), \dots, \text{current}_N(0)$  to position 0. (Note that most pointers are correctly aligned as desired given that  $\delta = n^{-(1/3+\varepsilon)}$  and thus,  $x_0$  is not deleted in most traces and  $f_i(\text{current}_i(0)) = f_i(0) = 0$  for most  $i$ .) The way the pointers are updated is as follows: At each time step  $t$ , we let  $w_t$  be the majority element of the  $N$ -element multiset  $\{\mathbf{y}_{\text{current}_i(t)}^i\}_{i \in [N]}$ . For those traces  $\mathbf{y}^i$  with  $\mathbf{y}_{\text{current}_i(t)}^i = w_t$  (i.e., the bit of  $\mathbf{y}^i$  at the current pointer is the majority bit), we move the pointer to the right by 1, i.e. we set  $\text{current}_i(t+1) \leftarrow \text{current}_i(t) + 1$ ; otherwise the pointer stays the same, i.e., we set  $\text{current}_i(t+1) \leftarrow \text{current}_i(t)$ . Next we increment  $t$  and start the next round, repeating until  $t = n$  when **BMA** outputs the string  $(w_0, \dots, w_{n-1})$ .

For intuition we observe that if most of the pointers were aligned at the beginning of time step  $t$  (i.e.,  $f_i(\text{current}_i(t)) = t$  for most  $i \in [N]$ ), then  $w_t = x_t$  is indeed the next bit in  $x$ . Moreover, if  $\text{current}_i(t)$  is aligned and  $w_t = x_t$ , then moving  $\text{current}_i$  to the right by 1 is justified by noting that most likely  $x_{t+1}$  is not deleted in  $\mathbf{y}^i$  (with probability  $1 - \delta$ ), and when this happens  $f_i(\text{current}_i(t+1)) = t+1$  so  $\text{current}_i$  remains aligned at the beginning of the next time step.

In more detail, our analysis shows that when  $x$  does not contain an  $s$ -desert for any  $s \in \{0, 1\}^{\leq C}$  **BMA** maintains the following invariants at the beginning of time step  $t = 0, 1, \dots, n$ :

1. At time  $t$ , **BMA** has reconstructed  $x_0, \dots, x_{t-1}$  correctly as  $w_0, \dots, w_{t-1}$ .
2. For every trace  $\mathbf{y}^i$ ,  $i \in [N]$ , it holds that  $f_i(\text{current}_i(t)) \geq t$ .
3. Finally,  $\sum_{i \in [N]} (f_i(\text{current}_i(t)) - t) \leq 2N/C$ .

The intuitive meaning behind conditions (2) and (3) is as follows: while (2) says that the “original position” of  $\text{current}_i(t)$  never falls behind  $t$ , condition (3) ensures that on average, the original positions of these pointers do not surpass  $t$  by too much. In fact, since  $C$  is a large constant, most of the pointers are perfectly aligned, i.e., they satisfy  $f_i(\text{current}_i(t)) = t$ .

<sup>2</sup> It will be convenient for us to index a binary string  $x \in \{0, 1\}^\ell$  using  $[0 : \ell - 1]$  as  $x = (x_0, \dots, x_{\ell-1})$ .



We now discuss how the invariants (1), (2) and (3) are maintained. First, we observe that invariant (1) for time step  $t + 1$ , i.e.,  $w_t = x_t$ , follows immediately from (3) at time step  $t$ . Invariant (2) for time step  $t + 1$  follows almost immediately from (2) at  $t$  and  $w_t = x_t$ . (If  $f_i(\text{current}_i(t)) > t$ , then  $f_i(\text{current}_i(t + 1)) \geq f_i(\text{current}_i(t)) \geq t + 1$  given that both  $f_i$  and  $\text{current}_i$  are nondecreasing; if  $f_i(\text{current}_i(t)) = t$  is aligned at time step  $t$ , then  $w_t = x_t$  implies  $\text{current}_i(t + 1) = \text{current}_i(t) + 1$  and thus,  $f_i(\text{current}_i(t + 1)) \geq t + 1$ .) The main challenge is to show that invariant (3) is maintained. While this is not true for a general string  $x$ , we show that this holds with high probability (over  $\mathbf{y}^1, \dots, \mathbf{y}^N \sim \text{Del}_\delta(x)$ ) for any string  $x$  which does not have an  $s$ -desert for any  $s \in \{0, 1\}^{\leq C}$ . (We note here that the value of  $m$  is selected so as to satisfy  $m\delta \ll 1$ ; on the other hand, when we discuss the **FindEnd** procedure below, we will see that we also require  $m$  to satisfy  $m \gg \sqrt{\delta n}$ .)

In a nutshell, the main proof idea for (3) is to exploit the fact that when we draw  $\mathbf{y}^1, \dots, \mathbf{y}^N \sim \text{Del}_\delta(x)$ , with high probability they satisfy two properties: (i) for every  $\mathbf{y}^i$  and every subword of roughly  $C^2 m$  consecutive positions in the original string  $x$ , no more than  $C$  positions within the subword are deleted in the generation of  $\mathbf{y}^i$ ; (ii) for every subword of roughly  $m$  consecutive positions in  $x$ , the number of  $\mathbf{y}^i$  that have at least one deletion in the subword is no more than  $N/C^3$ . These two properties can be shown using straightforward probabilistic arguments by taking advantage of the aforementioned  $m\delta \ll 1$ . Using these two properties, a detailed (non-probabilistic) argument shows that **BMA** can reconstruct the string  $x$  with high probability if  $x$  contains no  $s$ -desert.

The above discussion sketches our argument that if the target string  $x$  does not have an  $s$ -desert, then **BMA** correctly reconstructs  $x$ . More generally, our arguments show that if  $x$  does have an  $s$ -desert, then **BMA** correctly reconstructs the prefix of  $x$  up to the position when an  $s$ -desert shows up: Let  $r$  be the first position in  $x$  that is “deep in an  $s$ -desert”; this is the first position in  $x$  such that  $x_{[r-m:r+m]}$ <sup>3</sup>, the length- $M$  subword of  $x$  centered at  $r$ , is an  $s$ -desert. Then **BMA** correctly reconstructs the prefix of  $x$  up to position  $r + m$ . Having reached such a position, it is natural to now ask – “how do we determine the *end* of this desert?”. This naturally leads us to the next procedure **FindEnd**.

## 2.2 Overview of FindEnd

Suppose that  $x$  has an  $s$ -desert with  $|s| = k \leq C$ , so **BMA** reconstructs the length- $(r + m + 1)$  prefix of  $x$ , where  $r$  is the first position that is “deep in the  $s$ -desert” (note that it is easy to determine the position  $r$  from the output of **BMA**). The algorithm **FindEnd** takes as input the prefix  $x_{[0:r+m]}$  of  $x$  and the location  $r$ , and its task is to compute the end of the  $s$ -desert: the first position  $\text{end} \geq r + m$  such that  $x_{\text{end}+1} \neq x_{\text{end}-k+1}$ . The **FindEnd** algorithm is rather involved but at a high level it consists of two stages: an initial *coarse estimation* of the end of the desert followed by *alignments* of traces from  $\text{Del}_\delta(x)$  with the end of the desert (using the coarse estimate).

**Coarse estimation:** The goal of the coarse estimation stage is to identify an integer  $\hat{\beta}$  that is close to  $(1 - \delta)\text{end}$ :  $|\hat{\beta} - (1 - \delta)\text{end}| \leq 2\sigma$ , where  $\sigma := \tilde{O}(\sqrt{\delta n}) \ll m$  is basically how far an entry  $x_i$  of  $x$  can deviate from its expected location  $(1 - \delta)i$  in a typical trace  $\mathbf{y} \sim \text{Del}_\delta(x)$ . Intuitively,  $\hat{\beta}$  is an estimation of the location of  $x_{\text{end}}$  in a trace  $\mathbf{y} \sim \text{Del}_\delta(x)$  that contains it, i.e., when  $x_{\text{end}}$  is not deleted in  $\mathbf{y}$ . To do this, we draw  $\alpha = O(1/\varepsilon)$  many traces  $\mathbf{y}^1, \dots, \mathbf{y}^\alpha \sim \text{Del}_\delta(x)$ . Roughly speaking, we split each trace  $\mathbf{y}^i$  into overlapping intervals of

<sup>3</sup> For a string  $x \in [0 : n - 1]$  integers  $0 \leq a < b \leq n - 1$ , we write  $x_{[a:b]}$  to denote the *subword*  $(x_a, x_{a+1}, \dots, x_b)$ .



length  $4\sigma$ . The first interval starts at  $(1 - \delta)r$  and each successive interval shifts to the right by  $\sigma$  (so it overlaps with the previous interval by  $3\sigma$ ). Since  $m \gg \sigma = \tilde{O}(\sqrt{\delta n})$  (which is one of the bottlenecks that requires  $\delta \ll n^{-1/3}$ ), the  $s$ -desert is unlikely to end before  $(1 - \delta)r$  in a trace  $\mathbf{y} \sim \text{Del}_\delta(x)$  and must end in one of constantly many intervals with high probability, by the choice of  $\sigma$ . To identify one such interval, we make the following observation. Let  $\text{Cyc}_s$  be the set of all  $k$ -bit strings that can be obtained as cyclic shifts of  $s$ . Given  $\text{end}$  as the end of the  $s$ -desert that starts at  $x_{r-m}$ , every  $k$ -bit subword of  $x_{[r-m:\text{end}]}$  is in  $\text{Cyc}_s$  but  $(x_{\text{end}-k+2}, \dots, x_{\text{end}+1}) \notin \text{Cyc}_s$ , and these  $k \leq C$  bits will most likely remain in a trace given the low deletion rate. This motivates us to look for the leftmost interval  $I^*$  such that in at least half of  $\mathbf{y}^1, \dots, \mathbf{y}^\alpha$ , it holds that  $\mathbf{y}_{I^*}^i$  contains a  $k$ -bit subword not in  $\text{Cyc}_s$ . We show that with high probability, setting  $\hat{\beta}$  to be the right endpoint of  $I^*$  gives us a coarse estimate of  $(1 - \delta)\text{end}$  up to an accuracy of  $\pm 2\sigma$ .

In addition to obtaining  $\hat{\beta}$ , the coarse estimation stage recovers the following  $8\sigma$ -bit subword of  $x$ :  $(x_{\text{end}-k+2}, \dots, x_{\text{end}-k+8\sigma+1})$ , which we will refer to as the *tail string* of the  $s$ -desert and denote by  $\text{tail} \in \{0, 1\}^{8\sigma}$ . To this end, we draw another  $\alpha = O(1/\varepsilon)$  fresh traces  $\mathbf{y}^1, \dots, \mathbf{y}^\alpha$  and examine the subword of each  $\mathbf{y}^i$  of length  $6\sigma$  centered at location  $\hat{\beta}$ . Each  $\mathbf{y}^i$  looks for the first  $k$ -bit subword in this interval that is not in  $\text{Cyc}_s$  and votes for its  $8\sigma$ -bit subword that starts at this non-cyclic shift as its candidate for  $\text{tail}$ . We show that with high probability, the string with the highest votes is exactly  $\text{tail}$ . (We note that both parts of this coarse estimation procedure require that with high probability, any fixed interval of length  $O(\sigma)$  in  $x$  does not get any deletions in a random trace, i.e.,  $\sigma\delta \ll 1$ . This follows from the two constraints  $m\delta \ll 1$  and  $m \gg \sigma$ .)

**Alignments:** Suppose the first stage succeeds in computing  $\hat{\beta}$  and  $\text{tail} \in \{0, 1\}^{8\sigma}$ . The second stage is based on a procedure called **Align** which satisfies two crucial criteria. These criteria are as follows: if **Align** is given an input trace  $\mathbf{y} \sim \text{Del}_\delta(x)$ , then (a) with *fairly high* probability (by which we mean  $1 - n^{-\Theta(\varepsilon)}$  for the rest of the overview) it returns a location  $\ell$  in  $\mathbf{y}$  such that  $\mathbf{y}_\ell$  corresponds to  $x_{\text{end}}$  in  $x$ , and moreover (b) the expectation of  $\ell$  (over the randomness of  $\mathbf{y}$ ) is a “sharp estimate” of  $(1 - \delta)\text{end}$  that is accurate up to an additive  $\pm 0.1$  error.<sup>4</sup> To pin down the exact end of the  $s$ -desert, **FindEnd** simply draws  $\tilde{O}(n^{2/3-\varepsilon})$  many traces, runs **Align** on each of them and computes the average  $\hat{\ell}$  of the locations it returns. It is easy to show that rounding  $\hat{\ell}/(1 - \delta)$  to the nearest integer gives  $\text{end}$  with high probability.

The case when  $k = |s| = 1$  (so the desert is a long subword consisting either of all 0's or of all 1's) is significantly easier (and was implicitly handled in [2]), so in the following discussion we focus on the case when  $k = |s| \geq 2$  and the desert has a more challenging structure. For this case our **Align** procedure uses a new idea, which is that of a “signature.” A *signature* is a subword of  $x$ , denoted  $\text{sig}$ , of length between  $2k$  and  $8\sigma$  that starts at the same location  $x_{\text{end}-k+2}$  as  $\text{tail}$  (so  $\text{sig}$  is contained in  $\text{tail}$ , since  $|\text{tail}| = 8\sigma$ ) and either ends at a location  $d$  which is the smallest integer  $d \in [\text{end} + k + 1 : \text{end} + 8\sigma - k + 1]$  such that the  $k$ -bit subword that ends at  $d$  is not in  $\text{Cyc}_s$ , or has length  $8\sigma$  if no such  $d$  exists (in this case  $\text{sig}$  is the same as  $\text{tail}$ ). We remind the reader that the first  $k$ -bits of  $\text{tail}$ , and hence also of  $\text{sig}$ , is a string not in  $\text{Cyc}_s$ , and the same is true of the last  $k$  bits of  $\text{sig}$  if its length is less than  $8\sigma$ .

<sup>4</sup> We note that item (b) is not an immediate consequence of item (a). In more detail, the failure probability of (a) is roughly  $1/n^{\Theta(\varepsilon)}$ , but if when **Align** fails in (a) it returns a location that is inaccurate by  $\gg n^{\Theta(\varepsilon)}$  positions, then (b) would not follow from (a). Indeed significantly more effort is required in our analysis to ensure (b).

Given a trace  $\mathbf{y} \sim \text{Del}_\delta(x)$ , **Align** (roughly speaking) attempts to locate the image of  $x_{\text{end}}$  in  $\mathbf{y}$  by locating the image of **sig** within an interval in  $\mathbf{y}$  of length  $O(\sigma)$  around  $\hat{\beta}$ . In a bit more detail, it checks whether the restriction of  $\mathbf{y}$  to a certain interval  $J$  around  $\hat{\beta}$  is of the form  $w \circ \text{sig} \circ v$ , such that the first  $k$  bits of **sig** is the leftmost  $k$ -bit subword of  $\mathbf{y}_J$  that is not in  $\text{Cyc}_s$ . If  $\mathbf{y}$  does not satisfy this condition then **Align** discards that trace and outputs nil. We note that if the only goal of **Align** were to locate a position  $\ell$  in  $\mathbf{y}$  such that with fairly high probability  $\mathbf{y}_\ell$  corresponds to  $x_{\text{end}}$  (i.e. item (a) above), then in all other cases (i.e. whenever  $\mathbf{y}$  does satisfy the above condition) **Align** could return the index of the  $(k-1)$ -th bit of **sig** in  $\mathbf{y}_J$ . (By doing this, **Align** always returns the correct position whenever the subword of  $x$  of length  $O(\sigma)$  centered at **end** has no deletion in  $\mathbf{y}$  and  $x_{\text{end}}$  deviates from its expected location in a trace by at most  $\sigma$  in  $\mathbf{y}$ , which happens with probability  $O(\sigma\delta) = n^{-\Theta(\varepsilon)}$ .) However, it turns out that **Align** must proceed in a slightly (but crucially) different way in order to additionally satisfy item (b) above (i.e., have the expected value of its output locations provide an accurate “sharp estimate” of  $(1-\delta)\text{end}$ ). The actual execution of **Align** is that in the case when  $\mathbf{y}_J$  does satisfy the above condition, **Align** returns the index of the  $(k-1)$ -th bit of **sig** in  $\mathbf{y}_J$  with high probability and with some small remaining probability (the precise value of which depends on the location of **sig** within  $\mathbf{y}_J$ ), **Align** opts to still output nil. A detailed analysis, which we provide in Section 6.2.2, shows that this **Align** procedure satisfies both criteria (a) and (b) described above.

### 2.3 The overall algorithm

The overall algorithm works by alternately running **BMA** and **FindEnd**. It starts with **BMA**, which draws  $N = O(\log n)$  traces of  $x$  and returns the first position  $r$  in  $x$  that is deep in a desert as well as the prefix  $w = x_{[0:r+m]}$  of the target string  $x$ . Then the algorithm runs **FindEnd** to compute **end**, the right end of the desert. Note that the execution of **BMA** will misalign some small fraction of the traces it uses, but these errors do not affect **FindEnd** as **FindEnd** is run using fresh traces.

With **end** from **FindEnd**, the algorithm has now reconstructed the prefix  $x_{[0:\text{end}]}$  by extending  $x_{[0:r+m]}$ . Next the algorithm runs **BMA** again on  $N$  traces that are, ideally, drawn from  $x_{[\text{end}+1:n-1]}$ , in order to reconstruct the next segment of  $x$  until a new desert shows up (at which point the algorithm repeats). These traces are obtained by running the **Align** procedure used by **FindEnd** on  $N$  fresh traces  $\mathbf{y}^1, \dots, \mathbf{y}^N$  of  $x$ . Let  $\ell_i$  be the output of **Align** running on  $\mathbf{y}^i$ . As noted in (a) earlier, all but a small fraction of  $\ell_i$ 's are such that the desert ends at  $\mathbf{y}_{\ell_i}^i$  in  $\mathbf{y}^i$ . We then run **BMA** on  $\mathbf{z}^1, \dots, \mathbf{z}^N$ , where  $\mathbf{z}^i$  is the suffix of  $\mathbf{y}^i$  starting at  $\ell_i + 1$  for each  $i$ . Even though  $\mathbf{z}^1, \dots, \mathbf{z}^N$  are *not* exactly  $N$  fresh traces of  $x_{[\text{end}+1:n-1]}$  (since a small and arbitrary fraction of  $\mathbf{y}^i$  might be misaligned), **BMA** is able to succeed because of a crucial *robustness* property. This property is that the correctness guarantee of **BMA** holds even when a small and “adversarially” picked constant fraction of the  $N$  traces given to it are misaligned; intuitively, **BMA** enjoys this robustness because it works in each time step by taking a majority vote over its input traces, so as long as a substantial majority of the traces are correctly aligned, even a small constant fraction of adversarial traces cannot affect its correctness. The algorithm continues alternating between **BMA** and **FindEnd**, and is thereby able to reconstruct the entire target string  $x$ .

## 3 Preliminaries

**Notation.** Given a positive integer  $n$ , we write  $[n]$  to denote  $\{1, \dots, n\}$ . Given two integers  $a \leq b$  we write  $[a : b]$  to denote  $\{a, \dots, b\}$ . We write  $\ln$  to denote natural logarithm and  $\log$  to denote logarithm to the base 2. We denote the set of non-negative integers by  $\mathbb{Z}_{\geq 0}$ . We write “ $a = b \pm c$ ” to indicate that  $b - c \leq a \leq b + c$ .

**Subword.** It will be convenient for us to index a binary string  $x \in \{0, 1\}^n$  using  $[0 : n - 1]$  as  $x = (x_0, \dots, x_{n-1})$ . Given such a string  $x \in \{0, 1\}^n$  and integers  $0 \leq a \leq b \leq n - 1$ , we write  $x_{[a:b]}$  to denote the *subword*  $(x_a, x_{a+1}, \dots, x_b)$  of  $x$ . An  $\ell$ -*subword* of  $x$  is a subword of  $x$  of length  $\ell$ , given by  $(x_a, x_{a+1}, \dots, x_{a+\ell-1})$  for some  $a \in [0 : n - \ell]$ .

**Distributions.** When we use bold font such as  $\mathbf{D}, \mathbf{y}, \mathbf{z}$ , etc., it is to emphasize that the entity in question is a random variable. We write “ $\mathbf{x} \sim \mathcal{D}$ ” to indicate that random variable  $\mathbf{x}$  is distributed according to distribution  $\mathcal{D}$ .

**Deletion channel and traces.** Throughout this paper the parameter  $\delta : 0 < \delta < 1$  denotes the *deletion probability*. Given a string  $x \in \{0, 1\}^n$ , we write  $\text{Del}_\delta(x)$  to denote the distribution of the string that results from passing  $x$  through the  $\delta$ -deletion channel (so the distribution  $\text{Del}_\delta(x)$  is supported on  $\{0, 1\}^{\leq n}$ ), and we refer to a string in the support of  $\text{Del}_\delta(x)$  as a *trace* of  $x$ . Recall that a random trace  $\mathbf{y} \sim \text{Del}_\delta(x)$  is obtained by independently deleting each bit of  $x$  with probability  $\delta$  and concatenating the surviving bits.<sup>5</sup>

**A notational convention.** In several places we use sans serif font for names such as *tail* (which is a subword of the target string  $x$ ), *end* (which is a location in the target string  $x$ ), and so on. To aid the reader, whenever we use this font the corresponding entity is an “ $x$ -entity,” i.e. a location, subword, etc. that is associated with the source string  $x$  rather than with a trace of  $x$ .

## 4 The main algorithm

In this section we describe the main algorithm **Reconstruct**. We begin by giving a precise definition of the notion of an *s-desert*. To do this, here and throughout the paper we fix

$$C := \lceil 100/\varepsilon \rceil, \quad \text{and we recall that } m = n^{1/3} \quad \text{and} \quad M = 2m + 1.$$

► **Definition 2.** For  $s \in \{0, 1\}^{\leq C}$ , a binary string  $z \in \{0, 1\}^*$  is said to be an *s-desert* if  $z$  is a prefix of  $s^\infty$  and  $|z| \geq M$ . A string is said to be a *desert* if it is an *s-desert* for some  $s \in \{0, 1\}^{\leq C}$ . Given a string  $x \in \{0, 1\}^n$ , we say that a location  $i \in [0 : n - 1]$  is *deep* in a desert if the length- $M$  subword  $x_{[i-m:i+m]}$  centered at  $i$  is a desert. We say a string  $x$  has *no desert* if no subword of  $x$  is a desert (or equivalently, no location  $i \in [0 : n - 1]$  is deep in a desert in  $x$ ); otherwise we say that it has *at least one desert*.

### 4.1 The preprocessing step

Before stating the main algorithm, we first describe a simple preprocessing step.

► **Lemma 3.** There is a randomized algorithm **Preprocess** which satisfies the following with probability  $1 - n^{-\omega(1)}$  (over its internal randomness):

1. It outputs a string  $v \in \{0, 1\}^{n/2}$ .
2. For any unknown string  $x \in \{0, 1\}^n$ , given access to a sample from  $\text{Del}_\delta(x)$ , it can output a sample from  $\text{Del}_\delta(z)$ , where  $z = x \circ v$ , in linear time.
3. For any  $s \in \{0, 1\}^{\leq C}$ , the string  $v$  does not have a *s-desert*. Consequently, any desert in the string  $z = x \circ v$  ends at least  $n/2 - (2m + 1)$  bits before the end of  $z$ .

<sup>5</sup> For simplicity in this work we assume that the deletion probability  $\delta$  is known to the reconstruction algorithm. We note that it is possible to obtain a high-accuracy estimate of  $\delta$  simply by measuring the average length of traces received from the deletion channel.

■ **Algorithm 1** Algorithm **Reconstruct** for  $\delta = n^{-(1/3+\varepsilon)}$ .

---

**Input:** Length  $n$  of an unknown  $x \in \{0, 1\}^n$  and access to  $\text{Del}_\delta(x)$  where  $\delta = n^{-(1/3+\varepsilon)}$

**Output:** A string  $u$ , where the algorithm succeeds if  $u = x$

```

1 Set  $N := O(\log n)$ 
2 Draw  $N$  fresh traces  $z^1, \dots, z^N$  independently from  $\text{Del}_\delta(x)$ 
3 Run  $\text{BMA}(n, \{z^1, \dots, z^N\})$  and let  $w$  be its output
4 if  $w$  has no desert then return  $w$ 
5 else
6   | Let  $r$  be the first location that is deep in a desert in  $w$  and let  $u = w_{[0:r+m]}$ 
   // Main loop
7 for  $n/m$  rounds do
8   | Draw  $N$  fresh traces  $y^1, \dots, y^N$  independently from  $\text{Del}_\delta(x)$ 
9   | Run  $\text{FindEnd}(r, u, \{y^1, \dots, y^N\})$  and let  $b$  and  $\ell_i, i \in [N]$ , be its output
10  | Set  $r = b$  and extend  $u$  to be a string of length  $b$  such that  $u_{[r-m:b]}$  is a desert
11  | if  $b = n - 1$  then output "FAILURE"
12  | Let  $z^i$  be the suffix of  $y^i$  starting at  $y_{\ell_i+1}^i$  for each  $i \in [N]$ 
13  | Run  $\text{BMA}(n - b - 1, \{z^1, \dots, z^N\})$  and let  $w$  be its output
14  | if  $w$  has no desert then
15  |   | return  $u \circ w$ 
16  | else
17  |   | Let  $r^*$  be the first location that is deep in a desert in  $w$  and set  $r \leftarrow r + r^*$ 
18  |   | and  $u \leftarrow u \circ w_{[r^*+m]}$ 
19  |   | return  $u$  if  $u$  is of length  $n$ 
19 return  $u$ 

```

---

The algorithm chooses  $v$  to be a random string of length  $n/2$ . In order to obtain the original  $n$ -bit string  $x$  it suffices for us to reconstruct the  $(3n/2)$ -bit string  $z$ . The proof of correctness involves standard probabilistic arguments and is deferred to the full version.

For convenience of notation, we rename  $z$  as  $x$  and rename  $n$  to be the length of this string  $z$ , so we still have  $x = (x_0, \dots, x_{n-1})$ . Now  $x$  is an  $n$ -bit string that has the following property: any desert in  $x$  ends at least  $n/4$  bits before the right end of  $x$ . With this preprocessing accomplished, we now describe Algorithm **Reconstruct** in Algorithm 1.

## 4.2 The high level idea of the Reconstruct algorithm

At a high level the algorithm works as follows. It starts (lines 1-3) by drawing

$$N = O(\log n)$$

independent traces  $z^1, \dots, z^N$  from  $\text{Del}_\delta(x)$  and using them to run **BMA**. An important component of our analysis is the following new result about the performance of **BMA** (note that later we require, and will give, a more robust version of the theorem below; see Theorem 6):

► **Theorem 4.** Let  $\delta = n^{-(1/3+\varepsilon)}$  for some fixed constant  $\varepsilon > 0$ . Given  $N$  traces drawn independently from  $\text{Del}_\delta(x)$  for some unknown string  $x \in \{0,1\}^n$ , BMA runs in  $\tilde{O}(n)$  time and returns a string  $w$  of length  $n$  with the following performance guarantees:

1. If  $x$  has no desert, then  $w = x$  with probability at least  $1 - 1/n^2$ ;
2. If  $x$  has at least one desert, then  $w$  and  $x$  share the same  $(r + m + 1)$ -bit prefix with probability at least  $1 - 1/n^2$ , where  $r$  is the first location that is deep in a desert of  $x$ .

Let  $w$  be the string BMA returns. By Theorem 4, we have the following two cases:

1. If  $w$  has no desert, then also  $x$  has no desert and the algorithm can just return  $w$  (line 4);
2. If  $w$  has at least one desert, then writing  $r$  to denote the first location that is deep in a desert in  $w$ , it is safe to assume that  $w_{[0:r+m]} = x_{[0:r+m]}$  and  $r$  is also the first location that is deep in a desert in  $x$  (line 6).

Suppose that we are in the second case with  $w_{[0:r+m]} = x_{[0:r+m]}$ . Then  $w_{[r-m:r+m]}$  is an  $s$ -desert for some string  $s \in \{0,1\}^k$  of some length  $k \leq C$ . We let  $s$  be the shortest such string and let  $k$  be its length (so if  $w_{[r-m:r+m]}$  were, for example, a subword of the form 001001001001... of length a multiple of 12, we would take  $s = 001$  and  $k = 3$ ).

Next (lines 8-9) we run **FindEnd** to figure out where this repetition of  $s$  ends in  $x$ . We use **end** to denote the end of the desert, where  $\text{end} \geq r + m$  is the smallest integer such that  $x_{\text{end}+1} \neq x_{\text{end}-k+1}$ . By the preprocessing step we may assume that **end** exists and satisfies  $\text{end} \leq 3n/4$ . (We note that **FindEnd** has access to  $\text{Del}_\delta(x)$  to draw fresh traces by itself; we send  $N$  fresh traces  $\mathbf{y}^1, \dots, \mathbf{y}^N$  to **FindEnd** so that it can help align them to the end of the desert, which are used to run BMA later.) The performance guarantee for **FindEnd** is given below:

► **Theorem 5.** Let  $\delta = n^{-(1/3+\varepsilon)}$  for some fixed constant  $\varepsilon > 0$ . There is an algorithm **FindEnd** with the following input and output:

- **Input:** (i) a location  $r \in [0 : 3n/4]$ , (ii) a string  $u \in \{0,1\}^{r+m+1}$ , (iii) a multiset of strings  $\{\mathbf{y}^1, \dots, \mathbf{y}^N\}$  from  $\{0,1\}^{\leq n}$  where  $N = O(\log n)$ , and (iv) sample access to  $\text{Del}_\delta(x)$  for some unknown string  $x \in \{0,1\}^n$ .
- **Output:** An integer  $b$ , and an integer  $\ell_i$  for each  $i \in [N]$ .

The algorithm **FindEnd** draws  $\tilde{O}(n^{2/3-\varepsilon})$  many independent traces from  $\text{Del}_\delta(x)$ , runs in  $O(n^{5/3})$  time and has the following performance guarantee. Suppose that  $r$  is the first location that is deep in some desert of  $x$ ;  $u = x_{[0:r+m]}$ ; the unknown **end** of the desert to which  $x_r$  belongs is at most  $3n/4$ ; and  $\mathbf{y}^1 = \mathbf{y}^1, \dots, \mathbf{y}^N = \mathbf{y}^N$  are independent traces drawn from  $\text{Del}_\delta(x)$ . Then the integers  $b$  and  $\ell_i$  that **FindEnd** outputs satisfy the following properties with probability at least  $1 - 1/n^2$ :  $b = \text{end}$ , and  $\ell_i = \text{last}(\mathbf{y}^i)$  for at least 0.9 fraction of  $i \in [N]$ . Here  $\text{last}(y)$  for a trace  $y$  denotes the location  $\ell$  in  $y$  such that  $y_\ell$  corresponds to the last bit of  $x_{[0:\text{end}]}$  that survives in  $y$  (and we set  $\text{last}(y) = -1$  by default if all of  $x_{[0:\text{end}]}$  gets deleted in  $y$ ).

Line 9 runs **FindEnd** with fresh independent traces  $\mathbf{y}^1, \dots, \mathbf{y}^N$  drawn from  $\text{Del}_\delta(x)$ . By Theorem 5, with high probability **FindEnd** returns the correct location  $b = \text{end}$ , from which we can then recover  $x_{[0:b]}$  as the unique extension of  $w_{[0:r+m]}$  in which the pattern  $s$  keeps repeating until (and including) location  $b$ . Moreover, we have from Theorem 5 that, for at least a 9/10-fraction of all  $i \in [N]$ , the suffix  $\mathbf{z}^i$  of  $\mathbf{y}^i$  starting from  $\mathbf{y}_{\ell_i+1}^i$  is a trace drawn from  $\text{Del}_\delta(x_{[b+1:n-1]})$ . We further note that our preprocessing ensures  $b \leq 3n/4$  and thus, the algorithm does not halt on line 11.

To continue, we would like to run BMA again on  $\mathbf{z}^1, \dots, \mathbf{z}^N$  (the suffixes of  $\mathbf{y}^1, \dots, \mathbf{y}^N$ ) to recover  $x_{[b+1:n-1]}$  (or a prefix of  $x_{[b+1:n-1]}$  if it contains a desert). However, observe that now we need BMA to be robust against some noise in its input traces because by Theorem 5, up to

$1/10$  of  $z^1, \dots, z^N$  might have been obtained from an incorrect alignment of  $y^1, \dots, y^N$ . Thus we require the following more robust performance guarantee from **BMA**, given by Theorem 6 below. (To state this we need a quick definition: we say two multisets of strings of the same size are  $\eta$ -close if one can be obtained from the other by substituting no more than  $\eta$ -fraction of its strings. One should also consider  $x'$  in the statement below as  $x_{[b+1:n-1]}$  and  $n'$  as  $n - b - 1$ .)

► **Theorem 6.** *Let  $\delta = n^{-(1/3+\varepsilon)}$  for some fixed constant  $\varepsilon > 0$ . Suppose  $\tilde{z}^1, \dots, \tilde{z}^N$  are  $N$  independent traces drawn from  $\text{Del}_\delta(x')$  for some unknown string  $x' \in \{0, 1\}^{n'}$  with  $n' \leq n$ . The following holds with probability at least  $1 - 1/n^2$  over the randomness of  $\tilde{z}^1, \dots, \tilde{z}^N \sim \text{Del}_\delta(x')$ :*

1. *If  $x'$  has no desert, then **BMA** running on  $n'$  and any multiset  $\{z^1, \dots, z^N\}$  that is  $(1/10)$ -close to  $\{\tilde{z}^1, \dots, \tilde{z}^N\}$  returns  $w = x'$ ;*
2. *If  $x'$  has at least one desert, then **BMA** running on  $n'$  and any multiset  $\{z^1, \dots, z^N\}$  that is  $(1/10)$ -close to  $\{\tilde{z}^1, \dots, \tilde{z}^N\}$  returns a string  $w$  that shares the same  $(r' + m + 1)$ -bit prefix with  $x'$ , where  $r'$  is the first location that is deep in a desert in  $x'$ .*

Given Theorem 6, we can indeed successfully run **BMA** on  $z^1, \dots, z^N$  and with high probability, it correctly recovers a prefix of  $x_{[b+1:n-1]}$  up to the first point deep in the next desert (if any exists), in which case the algorithm repeats (if there is no next desert, then with high probability **BMA** will correctly recover the rest of  $x$ ).

### 4.3 Correctness of Reconstruct

The case when  $x$  has no desert is handled by Theorem 6. Assuming that  $x$  has at least one desert, it follows from Theorem 6 that  $r, u$  together satisfy the following property with probability at least  $1 - 1/n^2$  at the beginning of the main loop (lines 7-18):  $r$  is the first location that is deep in a desert in  $x$  and  $u = x_{[0:r+m]}$ . This gives the base case for the following invariant that the algorithm maintains with high probability:

**Invariant:** *At the beginning of each loop,  $r$  is the first location deep in some desert in  $x$  and  $u = x_{[0:r+m]}$ .*

Assume that the invariant is met at the beginning of the current loop. Let **end** denote the end of the current desert (i.e., the smallest value  $\text{end} \geq r + m$  such that  $x_{\text{end}+1} \neq x_{\text{end}-k+1}$ ; we observe that  $\text{end} \leq 3n/4$  always exists by the guarantee of the preprocessing step). Let  $y^1, \dots, y^N$  be fresh traces drawn at the beginning of this loop. For each  $i \in [N]$ , we write  $\tilde{z}^i$  to denote the suffix of  $y^i$  starting at  $\text{last}(y^i) + 1$ . Given that  $y^1, \dots, y^N \sim \text{Del}_\delta(x)$ ,  $\tilde{z}^1, \dots, \tilde{z}^N$  are indeed independent traces drawn from  $\text{Del}_\delta(x')$ , where  $x' = x_{[\text{end}+1:n-1]}$ . Then we note that, for the algorithm to deviate from the invariant in the current round, one of the following two events must hold for  $y^1, \dots, y^N$ :

1. **FindEnd**( $r, u, \{y^1, \dots, y^N\}$ ) fails Theorem 5; or
2.  $\{\tilde{z}^1, \dots, \tilde{z}^N\}$  fails Theorem 6, i.e., there is a multiset  $\{z^1, \dots, z^N\}$  that is  $(1/10)$ -close to  $\{\tilde{z}^1, \dots, \tilde{z}^N\}$  but **BMA**( $n - \text{end} - 1, \{z^1, \dots, z^N\}$ ) violates the condition in Theorem 6.

This is because whenever **FindEnd** succeeds, the strings  $\{z^1, \dots, z^N\}$  on which we run **BMA** on line 13 must be  $(1/10)$ -close to  $\{\tilde{z}^1, \dots, \tilde{z}^N\}$ . Theorem 5 ensures that item 1 happens with probability at most  $1/n^2$ ; Theorem 6 ensures that item 2 happens with probability at most  $1/n^2$ , given that  $\tilde{z}^1, \dots, \tilde{z}^N$  are independent traces from  $\text{Del}_\delta(x')$  as required in the assumption of Theorem 6.



---

**Algorithm 2** Algorithm BMA.

---

**Input:** A length  $n'$  and a multiset  $\{z^1, \dots, z^N\}$  of strings, each of length at most  $n'$   
**Output:** A string  $w = (w_0, \dots, w_{n'-1}) \in \{0, 1\}^{n'}$

- 1 For each  $i \in [N]$  pad each  $z^i$  to be a string  $u^i$  of length  $n'$  by adding 0's to the end
- 2 Set  $t = 0$  and  $\text{current}_i(t) = 0$  for each  $i \in [N]$
- 3 **while**  $t \leq n' - 1$  **do**
- 4     Set  $w_t \in \{0, 1\}$  to be the majority of the  $N$  bits  $u_{\text{current}_1(t)}^1, \dots, u_{\text{current}_N(t)}^N$
- 5     For each  $i \in [N]$ , set  $\text{current}_i(t+1)$  to  $\text{current}_i(t) + 1$  if  $u_{\text{current}_i(t)}^i = w_t$ ;  
       otherwise set  $\text{current}_i(t+1)$  to  $\text{current}_i(t)$
- 6     Increment  $t$ .
- 7 **return**  $w$ .

---

By a union bound, the invariant holds with high probability in every round given that we only repeat for  $n/m$  rounds. Finally, observe that we only need to repeat for  $n/m$  rounds to reconstruct the entire  $n$ -bit string  $x$ , since in each round the pointer  $r$  increases by at least  $2m$ .

This concludes the proof of correctness of **Reconstruct** and the proof of Theorem 1, modulo the proofs of Theorem 6 and Theorem 5. In the rest of the paper we prove those two theorems.

## 5 Improved analysis of the Bitwise Majority Algorithm: Proof of Theorem 6

The bitwise majority algorithm was first described and analyzed in [2]. The analysis given in [2] established that **BMA** successfully reconstructs any unknown source string  $x \in \{0, 1\}^n$  that does not contain any “long runs” (i.e., subwords of the form  $0^{n^{1/2+\varepsilon}}$  or  $1^{n^{1/2+\varepsilon}}$ ) provided that the deletion rate  $\delta$  is at most  $n^{-(1/2+\varepsilon)}$ . We describe the **BMA** algorithm in Algorithm 2. As the main result of this section we establish an improved performance guarantee for **BMA**. Our discussion and notation below reflects the fact that we will in general be running **BMA** “in the middle” of a string  $x$  for which we have already reconstructed a  $(b+1)$ -bit prefix of  $x$  (this is why Theorem 6 is stated in terms of a source string  $x'$  of length  $n' \leq n$ , which should be thought of as a suffix of  $x$ ). Our goal is to prove Theorem 6.

We break the proof of Theorem 6 into two steps (Lemma 8 and Lemma 14 below). For ease of exposition, in the rest of this section if  $x'$  has at least one desert then as stated in item (2) of the theorem, we let  $r'$  be the first location that is deep in a desert in  $x'$ . If  $x'$  has no desert, then we let  $r' = n' - m - 1$ . Note that with this definition of  $r'$ , it is guaranteed that there is no desert in  $x'_{[0:r'+m-1]}$  and the goal of **BMA** is to return a string that shares the same  $(r' + m + 1)$ -prefix with  $x$ .

Let  $R = 9N/10$ . We first prove in Lemma 8 that if a multiset of  $R$  traces  $Z = \{z^1, \dots, z^R\}$  of  $x'$  satisfies a certain sufficient “goodness” condition (see Definition 7 for details), then **BMA**( $n', Z$ ) not only returns a string  $w = (w_0, \dots, w_{n'-1}) \in \{0, 1\}^n$  that satisfies  $w_{[0:r'+m]} = x'_{[0:r'+m]}$  as desired but moreover, the bitwise majority during each of the first  $r' + m + 1$  rounds of **BMA** is “robust” in the following sense: for each one of those rounds, at least  $9R/10 = 81N/100$  of the  $R$  strings  $z^i$ 's agree with each other. This immediately implies that when  $Z$  satisfies this condition, adding any multiset of  $N/10$  strings to  $Z$  and running **BMA** on the resulting multiset of size  $N$  cannot affect the output of **BMA** during the first  $r' + m + 1$  rounds, so its output  $w$  still satisfies  $w_{[0:r'+m]} = x'_{[0:r'+m]}$ . The next lemma,



Lemma 14, shows that if  $\tilde{\mathbf{Z}} = \{\tilde{z}^1, \dots, \tilde{z}^N\}$  is a multiset of  $N$  traces drawn independently from  $\text{Del}_\delta(x')$  (as in the assumption part of Theorem 6), then with high probability *every*  $R$ -element subset of  $\tilde{\mathbf{Z}}$  satisfies the sufficient condition (Definition 7) for BMA to succeed robustly. Theorem 6 follows easily by combining Lemma 8 and Lemma 14. Due to lack of space, we defer most proofs in this section to the full version.

## 5.1 Notation for traces

We start with some useful notation for analyzing traces of  $x'$ . When a trace  $\mathbf{y}$  is drawn from  $\text{Del}_\delta(x')$  we write  $\mathbf{D}$  to denote the set of *locations deleted* when  $x'$  goes through the deletion channel, i.e.,  $\mathbf{D}$  is obtained by including each element in  $[0 : n' - 1]$  independently with probability  $\delta$ , and  $\mathbf{y}$  is set to be  $x'_{[0:n'-1] \setminus \mathbf{D}}$ . In the analysis of BMA when it is given as input  $R$  traces  $Z = \{z^1, \dots, z^R\}$ , our analysis will sometimes refer to the set  $D_i \subseteq [0 : n' - 1]$  of locations that was deleted when generating  $z^i$ .

Note that in the execution of BMA we pad each trace  $z^i$  to a string  $u^i$  of length  $n'$  by adding 0's to its end. In the rest of the section it will be convenient for us to view  $x'$  as a string of infinite length by adding infinitely many 0's to its end. We can then view each  $u^i$  as generated by first deleting the bits in  $D_i \subseteq [0 : n' - 1]$  from  $x'$  and taking the  $n'$ -bit prefix of what remains. This motivates the definition of the following map  $f_i : [0 : n' - 1] \rightarrow \mathbb{N}$  for each  $i \in [R]$ : For each  $j \in [0 : n' - 1]$ ,  $f_i(j)$  is set to be the unique integer  $k$  such that  $k \notin D_i$  and  $k - |D_i \cap [k - 1]| = j$ . In words,  $f_i(j)$  is simply the original location in  $x'$  of the  $j$ -th bit in the padded version  $u^i$  of  $z^i$ .

We specify some parameters that will be used in the rest of Section 5. Let  $C = \lceil 100/\varepsilon \rceil$  (so  $C$  should be thought of as a large absolute constant) and  $M = 2m + 1$  with  $m = n^{1/3}$ , and recall that by definition  $M$  is the shortest possible length of a desert.

## 5.2 BMA is robust on good sets of traces

The main result of this subsection is Lemma 8, which establishes that BMA is robustly correct in its operation on traces that satisfy a particular “goodness” condition given in Definition 7 below.

Let  $Z = \{z^1, \dots, z^R\}$  be a multiset of traces of  $x'$ . As described above we write  $u^i \in \{0, 1\}^{n'}$  to denote the 0-padded version of  $z^i$ ,  $D_i \subseteq [0 : n' - 1]$  to denote the set of locations that were deleted from  $x'$  to form  $z^i$ , and  $f_i$  to denote the map defined as above for each  $i \in [R]$ . We introduce the following condition for  $Z$  and then prove Lemma 8:

- **Definition 7.** We say  $Z = \{z^1, \dots, z^R\}$  is good if the following two conditions hold:
- (i) For every  $i \in [R]$  and every interval  $[\text{left} : \text{right}] \subset [0 : n' - 1]$  of length  $\text{right} - \text{left} + 1 = L_1 := 2C^2M$ , we have  $|D_i \cap [\text{left} : \text{right}]| \leq C$ .
  - (ii) For every interval  $[\text{left} : \text{right}] \subset [0 : n' - 1]$  of length  $\text{right} - \text{left} + 1 = L_2 := M + C + 1$ , the number of elements  $i \in [R]$  such that  $D_i \cap [\text{left} : \text{right}] \neq \emptyset$  is at most  $R/C^3$ .

Intuitively, (i) says that no interval of moderate length (note that this length  $2C^2M$  is polynomially less than  $1/\delta$ ) has “too many” deletions in it in any trace, whereas (ii) says that for every interval of moderate length (again polynomially less than  $1/\delta$ ), most of the  $R$  traces have no bit deleted within that interval.

Now we are ready to state Lemma 8:

- **Lemma 8.** Let  $Z = \{z^1, \dots, z^R\}$  be a good multiset of  $R$  traces of  $x'$ . Then the string  $w \in \{0, 1\}^{n'}$  that BMA( $n', Z$ ) outputs satisfies  $w_{[0:r'+m]} = x'_{[0:r'+m]}$ . Moreover, during each of the first  $r' + m + 1$  rounds of the execution of BMA, at least  $9R/10$  of the  $R$  bits in the majority vote taken in Step 4 of BMA agree with each other.

We start the proof of Lemma 8 by defining a map  $\text{distance}_i(t)$  for each  $z^i$  in  $Z$ . Recall that  $\text{current}_i(t)$  is the current location of the pointer into the padded trace  $u^i$  at the beginning of round  $t$  in BMA.<sup>6</sup> We let  $\text{position}_i(t) = f_i(\text{current}_i(t))$ , i.e. the original position in  $x'$  of the  $\text{current}_i(t)$ -th bit of  $u^i$ . Then  $\text{distance}_i(t)$  is defined as  $\text{distance}_i(t) = \text{position}_i(t) - t$ , the distance between  $t$  and  $\text{position}_i(t)$ . In Corollary 10 we will show that  $\text{distance}_i(t)$  is always nonnegative, and so it actually measures how many bits  $\text{position}_i(t)$  is ahead at round  $t$ . It may be helpful to visualize  $t$  and  $\text{position}_i(t)$  of a trace by writing down the source string  $x'$  with the deleted bits struck through, and having two arrows pointing to  $x'_t$  and  $x'_{\text{position}_i(t)}$ ; at the beginning of round  $t$ , the BMA algorithm tries to determine  $x'_t$  by looking at  $x'_{\text{position}_i(t)}$ . Intuitively, having  $\text{distance}_i(t) = 0$  means that the  $i$ -th trace was aligned properly at round  $t$ ; at the highest level, we establish Lemma 8 by showing that at least  $9R/10$  of the  $R$  traces have  $\text{distance}_i(t) = 0$ .

We state the following claim about how  $\text{current}_i(t)$ ,  $\text{position}_i(t)$  and  $\text{distance}_i(t)$  compare to their values at the beginning of round  $t-1$ , assuming that the prefix  $w_{[0:t-1]}$  of the output thus far matches  $x'_{[0:t-1]}$ .

▷ **Claim 9.** Let  $t$  be a positive integer such that  $w_{[0:t-1]} = x'_{[0:t-1]}$ . For each  $i \in [R]$ , we have

1. If  $x'_{\text{position}_i(t-1)} \neq x'_{t-1}$ , then  $\text{current}_i(t) = \text{current}_i(t-1)$ ,  $\text{position}_i(t) = \text{position}_i(t-1)$  and  $\text{distance}_i(t) = \text{distance}_i(t-1) - 1$ .
2. If  $x'_{\text{position}_i(t-1)} = x'_{t-1}$ , then  $\text{current}_i(t) = \text{current}_i(t-1) + 1$ ,  $\text{position}_i(t) = \text{position}_i(t-1) + \ell + 1$  and  $\text{distance}_i(t) = \text{distance}_i(t-1) + \ell$ , where  $\ell$  is the nonnegative integer such that  $\text{position}_i(t-1) + 1, \dots, \text{position}_i(t-1) + \ell \in D_i$  and  $\text{position}_i(t-1) + \ell + 1 \notin D_i$  (or equivalently,  $\ell = f_i(\text{current}_i(t)) - f_i(\text{current}_i(t-1)) - 1$ ).

We have the following useful corollary of Claim 9, which tells us that if  $w_{[0:t-1]} = x'_{[0:t-1]}$  then each  $\text{distance}_i(t) \geq 0$  (in other words, no trace can have “gotten behind” where it should be):

► **Corollary 10.** Let  $t$  be a positive integer such that  $w_{[0:t-1]} = x'_{[0:t-1]}$ . Then  $\text{distance}_i(t) \geq 0$  for all  $i \in [R]$ .

We prove three preliminary lemmas before proving Lemma 8. Recall that  $M = 2m + 1$  is the shortest possible length of a desert. Assuming  $w_{[0:t-1]} = x'_{[0:t-1]}$  for some  $t > M$ , the first lemma shows that if  $\text{distance}_i(t - M) = 0$  and no location of  $x'$  is deleted between  $t - M + 1$  and  $t$ , then  $\text{distance}_i(t)$  must stay at 0. (Note that this lemma holds for general  $M$  but we state it using  $M = 2m + 1$  for convenience since this is how it will be used later.) Intuitively, this says that if a length- $M$  subword of  $x'$  experiences no deletions, then a trace that is correctly aligned at the start of the subword will stay correctly aligned throughout the subword and at the end of the subword.

► **Lemma 11.** Suppose that  $w_{[0:t-1]} = x'_{[0:t-1]}$  for some  $t > M$ . Suppose that  $i \in [R]$  is such that  $\text{distance}_i(t - M) = 0$  and

$$D_i \cap [\text{position}_i(t - M) + 1 : \text{position}_i(t - M) + M] = D_i \cap [t - M : t] = \emptyset.$$

Then we have  $\text{distance}_i(t) = 0$ .

<sup>6</sup> Note that whereas  $\text{position}_i(\cdot)$  and  $\text{distance}_i(\cdot)$  refer to quantities defined in terms of the source string  $x$ ,  $\text{current}_i(\cdot)$  refers to a location in a trace string and not the source string.

In the second lemma, we assume  $t$  is such that  $M < t \leq r' + m + 1$  by the choice of  $r$ . We further assume that  $w_{[0:t-1]} = x'_{[0:t-1]}$  and  $0 < \text{distance}_i(t - M) \leq C$  for some  $i \in [R]$ . We show that under these assumptions, if the subword of length  $M$  in  $x'$  starting at  $\text{position}_i(t - M) + 1$  has no deletion, then  $\text{distance}_i(t) < \text{distance}_i(t - M)$ . Intuitively, this says that prior to a desert, if the length- $M$  subword of  $x'$  experiences no deletions and the alignment of a trace is only modestly ahead of where it should be at the start of the subword, then the alignment will improve by the end of the subword.

► **Lemma 12.** *Let  $M < t \leq r' + m + 1$  with  $w_{[0:t-1]} = x'_{[0:t-1]}$ . If  $0 < \text{distance}_i(t - M) \leq C$  for some  $i \in [R]$  and  $D_i \cap [\text{position}_i(t - M) + 1 : \text{position}_i(t - M) + M] = \emptyset$ , then we have  $\text{distance}_i(t) < \text{distance}_i(t - M)$ .*

Finally we use the two previous lemmas to show that if  $t \leq r' + m + 1$  and  $w_{[0:t-1]} = x'_{[0:t-1]}$ , then  $\text{distance}_i(t)$  must lie between 0 and  $C$ . Intuitively, this says that prior to a desert, the alignment of a trace will be at worst modestly ahead of where it should be.

► **Lemma 13.** *Let  $t \leq r' + m + 1$  and suppose that  $w_{[0:t-1]} = x'_{[0:t-1]}$ . Then  $\text{distance}_i(t) \leq C$  for all  $i \in [R]$ .*

**Proof of Lemma 8.** We prove by induction that for every positive integer  $t \leq r' + m + 1$ :

$$w_{[0:t-1]} = x'_{[0:t-1]} \quad \text{and} \quad \sum_{i \in [R]} \text{distance}_i(t) \leq \frac{2R}{C}. \quad (1)$$

It follows that every  $t \leq r' + m + 1$  satisfies  $|\{i \in [R] : \text{distance}_i(t) = 0\}| \geq R - 2R/C \geq 9R/10$  using  $C \geq 20$ . The details are deferred to the full version. ◀

### 5.3 Traces are good with high probability

To conclude the proof of Theorem 6 it remains to prove Lemma 14, which states that with high probability a random multiset of  $O(\log n)$  traces is such that every subset of 9/10 of the traces is good (recall Definition 7).

► **Lemma 14.** *Let  $\tilde{\mathbf{Z}} = \{\tilde{z}^1, \dots, \tilde{z}^N\}$  be a multiset of  $N = O(\log n)$  traces drawn independently from  $\text{Del}_\delta(x')$ . Then with probability at least  $1 - 1/n^2$ , every  $R$ -subset of  $\tilde{\mathbf{Z}}$  is good, where  $R = 9N/10$ .*

## 6 Finding the end of a desert: Proof of Theorem 5

In this section, we describe the algorithm **FindEnd**, which is used to determine the end of a desert in  $x$  using traces from  $\text{Del}_\delta(x)$ , and to align given traces with the end of the desert. (These aligned traces will then be used by **BMA** in the main algorithm.)

Let's recall the setting. Let  $x \in \{0, 1\}^n$  be the unknown string. **FindEnd** is given the first location  $r$  that is deep in some  $s$ -desert subword of  $x$ , for some string  $s \in \{0, 1\}^k$  with  $k \leq C$ . It is also given the prefix  $u = x_{[0:r+m]}$  of  $x$ . We will refer to the  $s$ -desert that contains  $r$  as the *current* desert. (Note that  $s$  can be easily derived from  $u$ .) The goal of **FindEnd** is to figure out the ending location of the current desert which we denote by **end**:

**end** is the smallest integer at least  $r + m$  such that  $x_{\text{end}+1} \neq x_{\text{end}-k+1}$ .

(Note that thanks to the preprocessing step **Preprocess**, we know that **end** exists and satisfies  $r + m \leq \text{end} \leq 3n/4$ .)

In addition to computing `end`, `FindEnd` is also given a multiset of  $N = O(\log n)$  traces  $y^1, \dots, y^N$  and needs to return a location  $\ell_i$  for each  $y^i$  such that most of them are correctly aligned to the end of the desert. Formally, we write  $\text{last}(y)$  for a trace  $y$  to denote the location  $\ell$  in  $y$  such that  $y_\ell$  corresponds to the last bit of  $x_{[0:\text{end}]}$  that survives in  $y$ ; we set  $\text{last}(y) = -1$  by default if all of  $x_{[0:\text{end}]}$  gets deleted. The second goal of `FindEnd` is to output  $\ell_i = \text{last}(y^i)$  for almost all  $y^i$  when they are drawn independently from  $\text{Del}_\delta(x)$ .

We present the algorithm `FindEnd` in Algorithm 3, where

$$\sigma := \lceil \sqrt{\delta n} \cdot \log n \rceil.$$

(Intuitively,  $\sigma$  provides a high-probability upper bound on how far a bit of  $x$  can deviate from its expected position in a trace  $y \sim \text{Del}_\delta(x)$ .) `FindEnd` consists of the following two main procedures:

1. We will refer to the  $8\sigma$ -bit string

$$\text{tail} := x_{\text{end}-k+2} x_{\text{end}-k+3} \cdots x_{\text{end}+8\sigma-k+1}$$

around the end  $x_{\text{end}}$  of the current desert as its *tail* string and denote it by  $\text{tail} \in \{0, 1\}^{8\sigma}$ . (Note that  $\text{end} + 8\sigma - k + 1 < n$  given that  $\text{end} \leq 3n/4$ .) The first procedure, `Coarse-Estimate`, will provide with high probability a *coarse estimate*  $\hat{\beta}$  (see Lemma 15) of the expected location  $(1 - \delta)\text{end}$  of the right end of the current desert in a trace of  $x$ . This procedure is described in Section 6.1.

2. With  $\hat{\beta}$  and  $\text{tail} \in \{0, 1\}^{8\sigma}$  in hand, the second procedure `Align` can help align a given trace with the right end of the current desert. Informally, running on a trace  $y \sim \text{Del}_\delta(x)$ , `Align` returns a position  $\ell$  such that with high probability over the randomness of  $y \sim \text{Del}_\delta(x)$ , it holds that  $\ell = \text{last}(y)$ . The performance guarantee of `Align` is given in Lemma 16. It may sometimes (with a small probability) return `nil`, meaning that it fails to align the given trace. This procedure is described in Section 6.2.

The algorithm `FindEnd` starts by running `Coarse-Estimate` to obtain a coarse estimate  $\hat{\beta}$  of  $(1 - \delta)\text{end}$  and the *tail* string (line 1). It then (line 2) runs `Align` on the given  $N$  traces  $y^i$  to obtain  $\ell_i$  for each  $i \in [N]$ . The second property of `FindEnd` in Theorem 5 about  $\ell_i$ 's follows directly from the performance guarantee of `Align`. To obtain a sharp estimate of `end`, `FindEnd` draws another set of  $\tilde{O}(n^{2/3-\varepsilon})$  traces  $z^i$  (line 3). It runs `Align` on each of them and uses the average of its outputs (discarding traces for which `Align` returns `nil`) to estimate  $(1 - \delta)\text{end}$  (lines 4-6). (It is clear that this average would be accurate to within  $\pm o(1)$  if `Align` always successfully aligned its input trace with the right end of the current desert; the actual performance guarantee of `Align` is weaker than this, but a careful analysis enables us to show that it is good enough for our purposes.)

## 6.1 The Coarse-Estimate procedure

Recall that  $\sigma = \lceil \sqrt{\delta n} \cdot \log n \rceil$ . Given  $r, u$  as specified earlier and sample access to  $\text{Del}_\delta(x)$ , the goal of `Coarse-Estimate` is to obtain an integer  $\hat{\beta}$  such that  $|\hat{\beta} - (1 - \delta)\text{end}| \leq 2\sigma$ . We will refer to such an estimate as a *coarse estimate* of  $(1 - \delta)\text{end}$ . In addition, `Coarse-Estimate` returns a string  $t$  that with high probability is exactly the tail string  $\text{tail} \in \{0, 1\}^{8\sigma}$ . This is done by drawing only  $O(1/\varepsilon)$  many traces.

► **Lemma 15.** *Let  $\delta = n^{-(1/3+\varepsilon)}$  with a fixed constant  $\varepsilon > 0$ . There is an algorithm `Coarse-Estimate` which takes the same two inputs  $r$  and  $u$  as in `FindEnd` and sample access to  $\text{Del}_\delta(x)$  for some unknown string  $x \in \{0, 1\}^n$ , and returns an integer  $\hat{\beta} \in [0 : n - 1]$  and a string  $t \in \{0, 1\}^{8\sigma}$ . It draws  $O(1/\varepsilon)$  traces from  $\text{Del}_\delta(x)$ , runs in time  $O(n)$  and has the following performance guarantee. Suppose that  $r$  and  $u$  satisfy the same conditions as in Theorem 5 with respect to  $x$ . Then with probability at least  $1 - 1/n^3$ , we have that  $t = \text{tail}$  and  $\hat{\beta}$  satisfies  $|\hat{\beta} - (1 - \delta)\text{end}| \leq 2\sigma$ .*

■ **Algorithm 3** Algorithm FindEnd.

---

**Input:**  $r \in [0 : 3n/4]$ ,  $u \in \{0, 1\}^{r+m+1}$ , a multiset  $\{y^1, \dots, y^N\}$  of  $N$  strings from  $\{0, 1\}^{\leq n}$  where  $N = O(\log n)$ , and sample access to  $\text{Del}_\delta(x)$  for some string  $x \in \{0, 1\}^n$ .

**Output:** An integer  $b \geq r + m$  and an integer  $\ell_i$  for each  $i \in [N]$ .

- 1 Run **Coarse-Estimate**( $r, u$ ), which returns an integer  $\hat{\beta}$  and a string  $t \in \{0, 1\}^{8\sigma}$ .
  - 2 For each  $i \in [N]$ , run **Align**( $\hat{\beta}, t, y^i$ ). If **Align** returns nil, set  $\ell_i = -1$ ; otherwise let  $\ell_i$  be the integer **Align** returns.
  - 3 Draw  $\gamma = O(n^{2/3-\varepsilon} \log^3 n)$  traces  $z^1, \dots, z^\gamma$  from  $\text{Del}_\delta(x)$ .
  - 4 For each  $i \in [\gamma]$ , run **Align**( $\hat{\beta}, t, z^i$ ) and let  $h_i$  be its output.
  - 5 Let  $\beta$  be the average of  $h_i$ 's that are not nil, and let  $b$  be the integer nearest to  $\beta/(1 - \delta)$ .
  - 6 Return  $b$ , and  $\ell_i$  for each  $i \in [N]$ .
- 

**Proof.** We start with the coarse estimate  $\hat{\beta}$ . Let  $\hat{r} = \lceil (1 - \delta)r \rceil$  and consider the following collection of overlapping intervals of positions in a trace of  $x$ :

$$\mathcal{I} := \left\{ [\hat{r} + j\sigma : \hat{r} + (j + 4)\sigma] : j \in \mathbb{Z}_{\geq 0} \right\}.$$

Note that each interval  $I$  contains  $4\sigma + 1$  positions. **Coarse-Estimate** draws  $\alpha = O(1/\varepsilon)$  traces  $y^1, \dots, y^\alpha$  from  $\text{Del}_\delta(x)$  and finds the leftmost interval  $I^* \in \mathcal{I}$  such that at least half of  $y^i$ 's satisfy the following property:  $y_{I^*}^i$  contains a  $k$ -bit subword that is not a cyclic shift of  $s$ . The algorithm then sets  $\hat{\beta}$  to be the right endpoint of  $I^*$ .

Finally, **Coarse-Estimate** recovers the tail string as follows. Let  $J'$  be the interval  $[\hat{\beta} - 3\sigma : \hat{\beta} + 3\sigma]$ . It draws another sequence of  $\alpha = O(1/\varepsilon)$  fresh traces  $y^1, \dots, y^\alpha$  from  $\text{Del}_\delta(x)$ . For each  $y^i$  it looks for the leftmost non-cyclic shift of  $s$  in  $y_{J'}^i$ . When such a non-cyclic shift exists, say  $y_\tau^i \cdots y_{\tau+k-1}^i$ ,  $y^i$  votes for the  $8\sigma$ -bit string  $y_\tau^i \cdots y_{\tau+8\sigma-1}^i$  as its candidate for the tail string. It then returns the  $8\sigma$ -bit string with the most votes.

Clearly, the running time of **Coarse-Estimate** is  $O(n)$  as the procedure consists of a linear scan over  $O(1/\varepsilon)$  traces. The proof of correctness is deferred to the full version. ◀

## 6.2 The Align procedure

We start with the performance guarantee of **Align**:

► **Lemma 16.** *Let  $\delta = n^{-(1/3+\varepsilon)}$  for some fixed constant  $\varepsilon > 0$ . There is an algorithm **Align** running in time  $O(n)$  with the following input and output:*

- **Input:** a number  $\hat{\beta} \in [0 : n - 1]$ , and strings  $t \in \{0, 1\}^{8\sigma}$ ,  $y \in \{0, 1\}^{\leq n}$ .
- **Output:** an integer  $\ell \in [0 : n - 1]$ , or nil.

*It has the following performance guarantee. Suppose  $x, u, r$  and end satisfy the hypothesis in Theorem 5,  $\hat{\beta}$  and  $t$  satisfy the conclusion of Lemma 15, and  $y = \mathbf{y} \sim \text{Del}_\delta(x)$  is a random trace. Then*

1. *Whenever **Align** returns an integer  $\ell$ , we have  $|\ell - (1 - \delta)\text{end}| \leq O(\sigma)$ .*
2. *With probability at least  $1 - \tilde{O}(n^{-3\varepsilon/2})$ , **Align** returns exactly  $\text{last}(\mathbf{y})$ ; and*
3. *Conditioned on **Align** not returning nil, the expectation of what **Align** returns is  $(1 - \delta)\text{end} \pm o(1)$ .*

### 6.2.1 Setup for the proof of Lemma 16

For the special case when  $k = |s| = 1$  (so the desert subword is of the form  $0^a$  or  $1^a$  for some  $a \geq M = 2n^{1/3} + 1$ ), the description of **Align** is relatively simple.

- **Description of Align for  $k = 1$ :** Let  $J := [\hat{\beta} - 3\sigma : \hat{\beta} + 3\sigma]$ . **Align** outputs nil if the string  $y_J$  contains no occurrence of  $\bar{b}$ ; if  $y_J$  does contain at least one occurrence of  $\bar{b}$  then **Align** outputs the location in  $J$  of the first occurrence of  $\bar{b}$ .

The proof of correctness is deferred to the full version. Now consider the general case when  $k \geq 2$ . Let  $\text{Cyc}_s$  be the set of all  $k$ -bit strings that can be obtained as cyclic shifts of  $s$ . The key notion behind **Align** is the idea of the “signature.” This is a subword of  $x$  of length at most  $8\sigma$  that starts at the same location  $x_{\text{end}-k+2}$  as **tail** (so it is contained in **tail**; we remind the reader that the first  $k$ -bits of **tail** is a string not in  $\text{Cyc}_s$ ). The signature ends at location  $d$  where  $d$  is the smallest integer  $d \in [\text{end} + k + 1 : \text{end} + 8\sigma - k + 1]$  such that the  $k$ -bit subword that ends at  $d$  is not in  $\text{Cyc}_s$ ; if no such  $d$  exists, the signature is taken to have length  $8\sigma$  and is the same as **tail**. (Alternatively, the signature is the shortest prefix of **tail** that contains a  $k$ -bit subword not in  $\text{Cyc}_s$  that does not use the first  $k$  bits; and it is set to **tail** if every  $k$ -bit subword of **tail** after removing the first  $k$  bits lies in  $\text{Cyc}_s$ .)

We will write **sig** to denote the signature string. We observe that  $2k \leq |\text{sig}| \leq 8\sigma$ , and that given the string **tail** it is algorithmically straightforward to obtain **sig**. Given **sig**, we say that a string  $z$  of length at most  $15\sigma + 1$  is in the *right form* if it can be written as

$$z = w \circ \text{sig} \quad (2)$$

where the leftmost  $k$ -bit subword in  $z$  that is not in  $\text{Cyc}_s$  is the first  $k$  bits of **sig**. The main motivation behind the definition of the signature is the following crucial lemma:

► **Lemma 17.** *Let  $s \in \{0, 1\}^k$  for some  $2 \leq k \leq C$ , and let  $z$  be a string of length at most  $15\sigma + 1$  that is in the right form. For  $\mathbf{y} \sim \text{Del}_\delta(z)$ , the probability that  $|\mathbf{y}| < |z|$  (so at least one deletion occurs) and  $\mathbf{y}$  is the prefix of a string in the right form is at most  $O(\delta)$ .*

The high-level idea is that a deletion is likely to create an additional disjoint  $k$ -bit subword in  $\mathbf{y}$  that is not in  $\text{Cyc}_s$ , unless a deletion occurs in some  $O(k)$  specific places in  $z$  or two deletions are  $O(k)$  close to each other. This additional subword will help us argue that  $\mathbf{y}$  does not have the right form. The detailed proof is deferred to the full version.

### 6.2.2 Proof sketch of Lemma 16 when $k \geq 2$

- **Description of Align for  $k \geq 2$ :** Given a coarse estimate  $\hat{\beta}$  (such that  $|\hat{\beta} - (1 - \delta)\text{end}| \leq 2\sigma$ ),  $\text{sig} \in \{0, 1\}^{\leq 8\sigma}$ , and a trace  $y$ , **Align** checks if the restriction of  $y$  to the interval  $J := [\hat{\beta} - 3\sigma : \hat{\beta} + 12\sigma]$  has a prefix in the right form (see Equation (2)), i.e.,

$$y_J = w \circ \text{sig} \circ v \quad (3)$$

so that the first  $k$  bits of **sig** is the leftmost  $k$ -bit subword of  $y_J$  not in  $\text{Cyc}_s$ . If  $y_J$  is not of this form **Align** returns nil. If  $y_J$  is of this form and **sig** ends at location  $L \in [0 : 15\sigma]$  in  $y_J$ , **Align** returns the index of the  $(k - 1)$ -th bit of **sig** (i.e.,  $\text{sig}_{k-2}$  which intuitively should correspond to  $x_{\text{end}}$ ) in  $y$  with probability

$$p_L := (1 - \delta)^{15\sigma - L}, \quad (4)$$

and with the remaining probability returns nil. Note that

$$(1 - \delta)^{15\sigma - L} = 1 - O(\delta\sigma) = 1 - \tilde{O}(n^{3\varepsilon/2}), \quad \text{for all } L \in [0 : 15\sigma],$$

so **Align** only returns nil with probability  $o(1)$  when  $y_J$  is of the form Equation (3).



**Discussion.** The main subtlety in the definition of **Align** is the “discounting probability” given by Equation (4), which plays an important role in ensuring that the location returned by **Align** (conditioned on **Align** not returning nil) is sufficiently close in expectation to the correct location. The proof of correctness of **Align** is deferred to the full version.

### 6.3 Proof of Theorem 5

**Proof of Theorem 5.** The proof follows from the guarantees in Lemma 15 and Lemma 16, using standard concentration bounds. First, we have from Lemma 15 that the output  $(\hat{\beta}, t)$  of **Coarse-Estimate** satisfies  $|\hat{\beta} - (1 - \delta)\text{end}| \leq 2\sigma$  and  $t = \text{tail}$  with probability  $1 - O(1)/n^3$ . Assume that this holds for the rest of the proof.

By Lemma 16, with probability at least  $1 - \tilde{O}(n^{-3\epsilon/2})$  **Align** returns an integer  $\ell_i$  (and not nil), and  $\ell_i = \text{last}(\mathbf{y}^i)$ , for each  $i \in [N]$ . Now, the Chernoff bound (additive form) implies that  $\ell_i = \text{last}(\mathbf{y}^i)$  for at least 0.9 fraction of  $i \in [N]$  with probability  $1 - \exp(-\Omega(N)) \geq 1 - 1/n^3$ , where we choose the hidden constant in  $N = O(\log n)$  to be sufficiently large.

It remains to show that  $b = \text{end}$  with probability at least  $1 - 1/n^3$ . Recalling step 4 of **FindEnd**, let  $G \subset [\gamma]$  be the set of indices  $i$  for which  $h_i = \text{Align}(\hat{\beta}, t, \mathbf{z}^i) \neq \text{nil}$ . Using the same argument as above, we have that  $|G| \geq 0.9\gamma$  with probability  $1 - \exp(-\Omega(\gamma)) = 1 - \exp(-\tilde{\Omega}(n^{2/3-\epsilon}))$ . The guarantees in Lemma 16 imply that  $|\mathbf{E}[h_i | h_i \neq \text{nil}] - (1 - \delta)\text{end}| \leq o(1)$  and that the random variable  $h_i$  (conditioned on its not being nil) always lies in an interval of width  $O(\sigma)$  for all  $i \in G$ . Moreover,  $\{h_i\}_{i \in G}$  are independent random variables.

Let  $\beta = (1/|G|) \sum_{i \in G} h_i$  be the average of  $h_i$  over  $i \in G$ . By Hoeffding’s inequality and our choice of  $\gamma = O(n^{2/3-\epsilon} \log^3 n) = O(\sigma^2 \log n)$  (with a sufficiently large hidden constant),

$$\Pr[|\beta - \mathbf{E}[h_i | h_i \neq \text{nil}]| \geq 0.1] \leq \exp\left(-\Omega\left(\frac{\gamma}{\sigma^2}\right)\right) \leq \exp(-\Omega(\log n)) \leq 1/n^3.$$

By triangle inequality,  $|\beta - (1 - \delta)\text{end}| \leq 0.1$ , and so  $|\beta/(1 - \delta) - \text{end}| \leq 0.2$ , with probability at least  $1 - 1/n^3$ . Hence, the integer  $b$  closest to  $\beta/(1 - \delta)$  is **end**, which implies **FindEnd** returns **end** with probability at least  $1 - 1/n^2$  (by union bound over all the failure probabilities).

Finally, the runtime of **FindEnd** is dominated by the final procedure to compute  $b$ . Since each run of **Align** on a trace takes  $O(n)$  and **Align** is run on  $\gamma \leq n^{2/3}$  traces, **FindEnd** runs in time  $O(n^{5/3})$ . This concludes the proof of Theorem 5.  $\blacktriangleleft$

---

### References

- 1 Alexandr Andoni, Constantinos Daskalakis, Avinatan Hassidim, and Sebastien Roch. Global alignment of molecular sequences via ancestral state reconstruction. *Stochastic Processes and their Applications*, 122(12):3852–3874, 2012.
- 2 T. Batu, S. Kannan, S. Khanna, and A. McGregor. Reconstructing strings from random traces. In *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2004*, pages 910–918, 2004.
- 3 Z. Chase. New lower bounds for trace reconstruction. *CoRR*, abs/1905.03031, 2019. [arXiv:1905.03031](#).
- 4 Anindya De, Ryan O’Donnell, and Rocco A. Servedio. Optimal mean-based algorithms for trace reconstruction. In *Proceedings of the 49th ACM Symposium on Theory of Computing (STOC)*, pages 1047–1056, 2017.
- 5 N. Holden and R. Lyons. Lower bounds for trace reconstruction. *CoRR*, abs/1808.02336, 2018. [arXiv:1808.02336](#).
- 6 Nina Holden, Robin Pemantle, and Yuval Peres. Subpolynomial trace reconstruction for random strings and arbitrary deletion probability. *CoRR*, abs/1801.04783, 2018. [arXiv:1801.04783](#).



- 7 Nina Holden, Robin Pemantle, Yuval Peres, and Alex Zhai. Subpolynomial trace reconstruction for random strings and arbitrary deletion probability. *CoRR*, abs/1801.04783, 2020. [arXiv:1801.04783](#).
- 8 T. Holenstein, M. Mitzenmacher, R. Panigrahy, and U. Wieder. Trace reconstruction with constant deletion probability and related results. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2008*, pages 389–398, 2008.
- 9 V. V. Kalashnik. Reconstruction of a word from its fragments. *Computational Mathematics and Computer Science (Vychislitel'naya matematika i vychislitel'naya tekhnika)*, Kharkov, 4:56–57, 1973.
- 10 Vladimir Levenshtein. Efficient reconstruction of sequences. *IEEE Transactions on Information Theory*, 47(1):2–22, 2001.
- 11 Vladimir Levenshtein. Efficient reconstruction of sequences from their subsequences or supersequences. *Journal of Combinatorial Theory Series A*, 93(2):310–332, 2001.
- 12 Andrew McGregor, Eric Price, and Sofya Vortnikova. Trace reconstruction revisited. In *Proceedings of the 22nd Annual European Symposium on Algorithms*, pages 689–700, 2014.
- 13 Michael Mitzenmacher. A survey of results for deletion channels and related synchronization channels. *Probability Surveys*, 6:1–33, 2009.
- 14 Fedor Nazarov and Yuval Peres. Trace reconstruction with  $\exp(o(n^{1/3}))$  samples. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017*, pages 1042–1046, 2017.
- 15 Lee Organick, Siena Dumas Ang, Yuan-Jyue Chen, Randolph Lopez, Sergey Yekhanin, Konstantin Makarychev, Miklos Z Racz, Govinda Kamath, Parikshit Gopalan, Bichlien Nguyen, et al. Random access in large-scale dna data storage. *Nature biotechnology*, 36(3):242, 2018.
- 16 S.M. Hossein Tabatabaei Yazdi, Ryan Gabrys, and Olga Milenkovic. Portable and error-free DNA-based data storage. *Scientific Reports*, 7(1):5011, 2017.

# Metrical Service Systems with Transformations

**Sébastien Bubeck**

Microsoft Research, Redmond, WA, USA  
sebubeck@microsoft.com

**Niv Buchbinder**

Tel Aviv University, Israel  
niv.buchbinder@gmail.com

**Christian Coester**

CWI, Amsterdam, The Netherlands  
christian.coester@cwi.nl

**Mark Sellke**

Stanford University, CA, USA  
msellke@stanford.edu

---

## Abstract

We consider a generalization of the fundamental online metrical service systems (MSS) problem where the feasible region can be transformed between requests. In this problem, which we call T-MSS, an algorithm maintains a point in a metric space and has to serve a sequence of requests. Each request is a map (transformation)  $f_t: A_t \rightarrow B_t$  between subsets  $A_t$  and  $B_t$  of the metric space. To serve it, the algorithm has to go to a point  $a_t \in A_t$ , paying the distance from its previous position. Then, the transformation is applied, modifying the algorithm's state to  $f_t(a_t)$ . Such transformations can model, e.g., changes to the environment that are outside of an algorithm's control, and we therefore do not charge any additional cost to the algorithm when the transformation is applied. The transformations also allow to model requests occurring in the  $k$ -taxi problem.

We show that for  $\alpha$ -Lipschitz transformations, the competitive ratio is  $\Theta(\alpha)^{n-2}$  on  $n$ -point metrics. Here, the upper bound is achieved by a deterministic algorithm and the lower bound holds even for randomized algorithms. For the  $k$ -taxi problem, we prove a competitive ratio of  $\tilde{O}((n \log k)^2)$ . For chasing convex bodies, we show that even with contracting transformations no competitive algorithm exists.

The problem T-MSS has a striking connection to the following deep mathematical question: Given a finite metric space  $M$ , what is the required cardinality of an extension  $\tilde{M} \supseteq M$  where each partial isometry on  $M$  extends to an automorphism? We give partial answers for special cases.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Online algorithms

**Keywords and phrases** Online algorithms, competitive analysis, metrical task systems,  $k$ -taxi

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.21

**Funding** *Niv Buchbinder*: Supported by Israel Science Foundation Grant 2233/19 and United States - Israel Binational Science Foundation Grant 2018352.

*Christian Coester*: Supported by NWO VICI grant 639.023.812 of Nikhil Bansal. Part of this work was carried out while he was visiting Microsoft Research, Redmond.

*Mark Sellke*: Supported by NSF graduate research fellowship and Stanford graduate fellowship.

## 1 Introduction

*Metrical Service Systems (MSS)* [13] is a fundamental online framework unifying countless problems. It also has a central role in our understanding of online computation and competitive analysis in general. In this problem we are given a metric space  $(M, d)$ . The points of the metric represent possible states/configurations where an algorithm can serve requests; the distance between the states represents the cost of moving from one configuration to



© Sébastien Bubeck, Niv Buchbinder, Christian Coester, and Mark Sellke;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 21; pp. 21:1–21:20

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

another. Each request consists of a subset of feasible states and the algorithm must serve the request by moving to one of these states. The cost of the algorithm for a sequence of requests is simply the total movement cost.

In the closely related problem of *Metrical Task Systems (MTS)* [8], each request is a cost function  $c_t: M \rightarrow \mathbb{R}_+ \cup \{\infty\}$ . The algorithm can move to any point  $x_t$ , paying the movement cost as well as service cost  $c_t(x_t)$ . Note that MSS is equivalent to the special case of MTS where cost functions only take values 0 and  $\infty$ , which already captures the essential difficulty of MTS.<sup>1</sup> For deterministic algorithms, the competitive ratio on any  $n$ -point metric is  $n - 1$  for MSS [19] and  $2n - 1$  for MTS [8]. For randomized algorithms, it lies between  $O(\log^2 n)$  [9, 15] and  $\Omega(\log n / \log \log n)$  [5, 6], and tight bounds of  $\Theta(\log n)$  are known for some metrics. The MSS/MTS framework captures various central online problems such as paging,  $k$ -server, convex body chasing, layered graph traversal, etc. The competitive ratio of MSS usually serves as a first upper bound for the performance achievable for these special cases.

However, MSS fails to capture more dynamic environments in which configuration changes that are outside of the algorithm's control may occur. For example, new resources or constraints may appear/disappear and modify the configurations. To capture such changes we propose an extension for MSS that allows transformations over the configuration space. For example, we may model the  $k$ -taxi problem<sup>2</sup> by considering the possible configurations of taxis in the metric. A movement of a taxi from the start to the destination of a request simply corresponds to a transformation that maps any configuration that contains a taxi at the start to a configuration with an additional taxi at the destination and one less taxi at the start. As these changes are dictated to any solution, it is reasonable not to account any cost for these changes both for the algorithm and for the offline (benchmark) solution. In this work we initiate the study of *Metrical Service Systems with Transformations (T-MSS)*, a generalization of the standard Metrical Service Systems (MSS) problem. As before, we are given a metric space  $(M, d)$ . In each round  $t$ , we get a function (transformation)  $f_t: A_t \rightarrow B_t$  that maps a subspace  $A_t \subseteq M$  of feasible states to a subspace  $B_t \subseteq M$ . If  $b_{t-1} \in M$  is the state of the algorithm before the request  $f_t$  arrives, it has to choose one of the feasible states  $a_t \in A_t$  and pays movement cost  $d(b_{t-1}, a_t)$ . The new state of the algorithm is then  $b_t := f_t(a_t)$ . The classical MSS problem is thus the special case of T-MSS where  $f_t$  is the identity function on the set of feasible states at time  $t$ . In T-MSS we allow in addition to identity transformations also more complex transformations. The high level question we ask is:

► **Open problem 1.** *What is the competitive ratio of the Metrical Service Systems with Transformations problem for families of metric spaces and allowable transformations?*

## 1.1 Our results and techniques

We give partial answers to the above question, obtaining upper and lower bounds on the competitiveness for several interesting families of metric spaces and transformations. The most general family of transformations we study are  $\alpha$ -Lipschitz transformations. Our main result is the following pair of (almost matching) upper and lower bounds for general metric spaces and  $\alpha$ -Lipschitz transformations.

<sup>1</sup> Our results extend easily to the case where MTS requests are allowed, but we will stick to the MSS view for the sake of simplicity.

<sup>2</sup> In this problem, introduced by [17], there are  $k$  taxis in a metric space. Each request is a pair of two points, representing the start and destination of a travel request by a passenger. Serving a request is done by selecting a taxi that travels first to its start and then its destination. In the hard version of the problem, the cost is defined as the total distance traveled by the taxis *without carrying a passenger*.

► **Theorem 1.** *There exists a deterministic  $2 \max\{2(\alpha + 1), 6\}^{n-2}$ -competitive algorithm for T-MSS with  $\alpha$ -Lipschitz transformations on any  $n$ -point metric space. Any algorithm for T-MSS with  $\alpha$ -Lipschitz transformations has competitive ratio at least  $\min\{\alpha + 1, \alpha^2\}^{n-2}$ , even with randomization.*

Although our results show an exponential lower bound (and an exponential upper bound) for any  $\alpha > 1$ , they do not rule out linear/polynomial or with randomization even polylogarithmic competitive ratios when the transformations are 1-Lipschitz (contractions). Resolving the competitive ratio for this important family of transformations is one of the most interesting remaining open questions. On the other hand, we show that even restricting the transformations to be 1-Lipschitz is not always enough. In particular, when adding contraction transformations to the convex body chasing problem (that can be modeled as a special case of MSS on an infinite metric space) there exists no competitive algorithm even in the easier nested case. By contrast, with isometry transformations the competitive ratios  $O(d)$  for the unrestricted problem and  $O(\sqrt{d \log d})$  for the nested problem due to [20, 2, 10] remain unchanged as  $\mathbb{R}^d$  is ultrahomogeneous (see following discussion for formal definitions).

► **Theorem 2.** *There exists no online algorithm with finite competitive ratio for nested convex body chasing with contractions in the plane, even with randomization.*

As a byproduct of our results we also get a new competitive algorithm for the  $k$ -taxi problem, which can be modeled as a T-MSS with isometry transformations.

► **Theorem 3.** *There is a randomized  $O((n \log k)^2 \log n)$ -competitive algorithm for the  $k$ -taxi problem on  $n$ -point metrics.*

This result is better than the previous best bound of  $O(2^k \log n)$  whenever  $n$  is subexponential in  $k$  [14].

**Extending partial isometries.** As a basic tool to tackle T-MSS with general transformation, we study the problem when the allowable transformations are isometries. A map  $f: A \rightarrow B$  for subsets  $A, B \subseteq M$  is called a *partial isometry of  $M$*  if it is distance-preserving, i.e.,  $d(f(x), f(y)) = d(x, y)$ . A metric space  $M$  is called *ultrahomogeneous* if every partial isometry of  $M$  extends to an automorphism of  $M$ . Notice that on ultrahomogeneous metric spaces, the competitive ratio of T-MSS with isometry transformations is the same as the competitive ratio of MSS: Indeed, when a partial isometry  $f_t: A_t \rightarrow B_t$  arrives, let  $\hat{f}_t$  be its extension to an automorphism of  $M$ . In MSS, this request corresponds to the request  $A_t$  followed by a renaming of the points of the metric space according to  $\hat{f}_t$ . Clearly, isometric renaming of points does not affect the competitive ratio.

If  $M$  is not ultrahomogeneous, one could hope to extend  $M$  to a larger metric space  $\hat{M}$  such that every partial isometry of  $M$  extends to an automorphism of  $\hat{M}$ . In this case, we call  $\hat{M}$  a *weakly ultrahomogeneous extension* of  $M$ . For a family of metric spaces  $\mathcal{M}$  we define the *blow-up* as the supremum over all  $n$ -point metrics  $M \in \mathcal{M}$  – of the minimum cardinality of a weakly ultrahomogeneous extension  $\hat{M}$  of  $M$ . If the blow-up can be bounded as a function of  $n$  it allows us to apply the  $(n - 1)$ -competitive deterministic algorithm or the  $O(\log^2 n)$ -competitive randomized algorithm for MSS on the weakly ultrahomogeneous extension to get competitive algorithms. We study the blow-up also for restricted families of isometries. For example, we study swap isometries (defined on two points that are mapped to one another). We call a metric space *swap-homogeneous* if every swap extends to an automorphism. Swap-homogeneous metric spaces have the intuitive property that the metric space “looks” the same at each point. Similar notions of metric space homogeneity have also been studied in other contexts (see, e.g., [23]). The main question, which we consider to be of independent interest, is thus:

► **Open problem 2.** *What is the blow-up for interesting families of metric spaces and partial isometries?*

Some results on weakly ultrahomogeneous extensions already exist, in particular that every finite metric space has such an extension and it is of finite size [21, 22]. However, no bounds on the size of the extensions as a function of  $n$  are known, and hence these results do not yield any bounds on the blow-up. We refer the reader to Section 5 for further discussion. Some of the algorithms we design for T-MSS are using these new upper bounds that we prove on the blow-up. The following theorem summarizes the upper and lower bounds we obtain.

► **Theorem 4.** *The following bounds on the blow-up are tight.*

Family of metric spaces	Family of isometries	Blow-up
Ultrametrics	General	$2^{n-1}$
Ultrametrics: $k$ distinct non-zero distances	General	$\approx \left(\frac{n+k-1}{k}\right)^k$ <sup>3</sup>
Equally spaced points on a line	General	$2n - 2$
$(\{0, \dots, k\}^D, \text{weighted } \ell_1)$	Translations	$(2k)^D$
General metrics	Swaps	$2^{n-1}$

Using the direct reduction above, we may get directly some interesting results for T-MSS when transformations are isometries. For example, a randomized  $O(n^2)$ -competitive algorithm for ultrametrics and an  $O(\log^2 n)$ -competitive algorithm for equally spaced points on a line.

**The work function algorithm.** The *work function algorithm* (WFA) is a classical algorithm that achieves the optimal deterministic competitive ratio of  $n - 1$  for MSS on any  $n$ -point metric (see [7] for a discussion on its history). The algorithm extends naturally to T-MSS (see Section 2.1), and is a natural candidate algorithm to investigate. We prove that it is in fact optimal for several special cases of the problem. The following result for ultrahomogeneous ultrametrics is also used as part of our main algorithm for general metrics. We also prove that on general metrics, WFA has a superlinear competitive ratio even for isometry transformations (which are 1-Lipschitz). This may indicate that T-MSS has super linear competitive ratio even for isometry transformations.

- **Theorem 5.** *The work function algorithm (WFA) for T-MSS has competitive ratio*
- $n - 1$  on  $n$ -point ultrahomogeneous ultrametrics with 1-Lipschitz transformations.
  - $2n - 3$  on  $n$ -point metric spaces with swap transformations. No deterministic algorithm has competitive ratio better than  $2n - 3$  for this problem.
  - $\omega(n^{1.29})$  on some  $n$ -point metric space with isometry transformations, for each  $n$ .

## 1.2 Organization

In Section 2 we formally define T-MSS, and discuss the work function algorithm (WFA). In Section 3 we design an algorithm for T-MSS on general metrics with competitive ratio depending on the maximal Lipschitz constant of transformations, and prove an almost matching lower bound (Theorem 1). As part of the algorithm we also show that WFA is  $(n-1)$ -competitive for ultrahomogeneous  $n$ -point ultrametrics and 1-Lipschitz transformations, proving the first part of Theorem 5.

<sup>3</sup> The precise blow-up is  $(a+1)^b a^{k-b}$  where  $a = \lfloor \frac{n+k-1}{k} \rfloor$  and  $b = (n-1) \bmod k$ .

In Section 4 we show upper and lower bounds on the competitive ratio for several special cases of T-MSS. In Section 4.1 we show that there exists no online algorithm with finite competitive ratio for nested convex body chasing with contractions in the plane, even with randomization (Theorem 2). In Section 4.2 we design a new randomized algorithm for the  $k$ -taxi problem (Theorem 3). In Section 4.3 we show matching upper and lower bounds for swap transformations, proving the second part of Theorem 5. In Section 4.4 we show a superlinear lower bound on the competitiveness of WFA for isometry transformations, proving the third part of Theorem 5. Finally, in Section 5 we prove upper and lower bounds on the blow-up for several families of metric spaces and transformations (Theorem 4). These results are also used earlier in the proofs of Theorem 1 and Theorem 3.

## 2 Preliminaries

In *Metrical Service Systems with Transformations* (T-MSS), we are given a metric space  $(M, d)$  and an initial state  $b_0 \in M$ . We denote by  $n$  the number of points in the metric space. In each round  $t$ , we get a function (transformation)  $f_t: A_t \rightarrow B_t$  that maps a subset  $A_t \subseteq M$  of feasible states to a subset  $B_t \subseteq M$ . If  $b_{t-1} \in M$  is the state of the algorithm before  $f_t$  arrives, it has to choose one of the feasible states  $a_t \in A_t$  and pays movement cost  $d(b_{t-1}, a_t)$ . The new state of the algorithm is then  $b_t := f_t(a_t)$ . The classical MSS problem is thus a special case of T-MSS where  $f_t$  is the identity function on the set of feasible states at time  $t$ . In T-MSS we always allow the identity transformations (thereby ensuring that T-MSS is a generalization of MSS) and in addition more complex transformations. We study the following important families of transformations  $f: A \rightarrow B$  whose properties are defined below.

Family of transformations	Condition
$\alpha$ -Lipschitz	$\forall x, y \in A: d(f(x), f(y)) \leq \alpha \cdot d(x, y)$
1-Lipschitz (Contractions)	$\forall x, y \in A: d(f(x), f(y)) \leq d(x, y)$
Isometries	$\forall x, y \in A: d(f(x), f(y)) = d(x, y)$
Swaps	$A = \{a, b\}, f(a) = b, f(b) = a$
Translations	$M$ is subset of a vector space. $\forall x \in A: f(x) = x + v$ for a vector $v$

We also study several families of metric spaces. An important family of metric spaces are ultrametrics in which for every three points  $x, y, z \in M$ ,  $d(x, z) \leq \max\{d(x, y), d(y, z)\}$ . Ultrametric spaces may be viewed as the leaves of a rooted tree in which vertices with lowest common ancestor at level  $i$  have distance  $L_i$ , where  $0 < L_1 < L_2 < \dots < L_k$  are the possible distances (where  $k \leq n - 1$ ).

### 2.1 The work function algorithm for T-MSS

The *work function algorithm* (WFA) achieves the optimal deterministic competitive ratio of  $n - 1$  for MSS on any  $n$ -point metric. This algorithm is defined as follows: Denote by  $p_0$  the fixed initial state. For some request sequence and a state  $p \in M$ , let  $w_t(p)$  be the minimal cost to serve the first  $t$  requests and then end up at  $p$ . The function  $w_t$  is called the *work function* at time  $t$ . We denote by  $\mathcal{W}$  the set of all maps  $w: M \rightarrow \mathbb{R}_+$  that are 1-Lipschitz. Notice that every work function is in  $\mathcal{W}$ . The WFA for MSS is the algorithm that, at time  $t$ , goes to a feasible state  $p_t$  minimizing  $w_t(p_t) + d(p_t, p_{t-1})$ .

This algorithm extends naturally to T-MSS: Let  $f_t: A_t \rightarrow B_t$  be the  $t$ th transformation. Let  $w_t$  be defined as above and let  $w_t^-(p)$  be the minimal cost of serving the first  $t-1$  requests, then moving to some state in  $A_t$  and then moving to  $p$ . Let  $b_{t-1}$  be the state of the algorithm before time  $t$ . Upon the arrival of  $f_t$ , the WFA first goes to a state

$$a_t \in \arg \min_{a \in A_t} w_t^-(a) + d(a, b_{t-1})$$

and is then relocated to  $b_t := f_t(a_t)$ .

We say that a work function  $w \in \mathcal{W}$  is *supported* on a set  $S \subseteq M$  if  $w(x) = \min_{s \in S} w(s) + sx$  for each  $x \in M$ . The (unique) minimal such set  $S$  is called the *support* of  $w$ . Notice that  $w_t^-$  is supported on  $A_t$  and  $w_t$  on  $B_t$ .

The following lemma is a variant of a lemma that is ubiquitous in analyses of WFA for other problems [13], adapted here to T-MSS:

► **Lemma 6.** *Let  $M$  be a metric space. Suppose there is a map  $\Phi: \mathcal{W} \rightarrow \mathbb{R}_+$  such that for any  $x \in M$ , time  $t \geq 1$ ,  $w \in \mathcal{W}$ , and any sequence of work functions  $w_0, w_1^-, w_1, w_2^-, w_2, \dots$  arising for (a subclass of) T-MSS on  $M$ ,*

$$w_t^-(x) - w_{t-1}(x) \leq \Phi(w_t^-) - \Phi(w_{t-1}) \quad (1)$$

$$\Phi(w_t^-) \leq \Phi(w_t) \quad (2)$$

$$\Phi(w) \leq \rho_M \cdot \min_{p \in M} w(p) + C_M, \quad (3)$$

where  $\rho_M$  and  $C_M$  are constants depending only on  $M$ . Then WFA is  $(\rho_M - 1)$ -competitive for (this subclass of) T-MSS on  $M$ .

**Proof.** We have

$$\begin{aligned} w_t^-(b_{t-1}) &= \min_{a \in A_t} w_t^-(a) + d(a, b_{t-1}) \\ &= w_t^-(a_t) + d(a_t, b_{t-1}) \\ &= w_t(b_t) + d(a_t, b_{t-1}), \end{aligned} \quad (4)$$

where the first equation is by definition of  $w_t^-$  and the second equation by definition of  $a_t$ . The cost of WFA is

$$\begin{aligned} \text{cost}_{\text{WFA}} &= \sum_{t=1}^T d(b_{t-1}, a_t) \\ &= \sum_{t=1}^T (w_t^-(b_{t-1}) - w_t(b_t)) \\ &= \sum_{t=1}^T (w_t^-(b_{t-1}) - w_{t-1}(b_{t-1}) + w_{t-1}(b_{t-1}) - w_t(b_t)) \\ &\leq \Phi(w_T) - w_T(b_T) \\ &\leq (\rho_M - 1) \cdot \min_{p \in M} w_T(p) + C_M, \end{aligned}$$

where the second equation follows from (4), the first inequality uses (1), (2),  $\Phi(w_0) \geq 0$  and  $w_0(b_0) = 0$ , and the second inequality uses (3). Since  $\min_{p \in M} w_T(p)$  is the optimal offline cost, the lemma follows. ◀



### 3 Competitiveness for Lipschitz Transformations

In this Section we prove Theorem 1.

► **Theorem 1.** *There exists a deterministic  $2 \max\{2(\alpha + 1), 6\}^{n-2}$ -competitive algorithm for T-MSS with  $\alpha$ -Lipschitz transformations on any  $n$ -point metric space. Any algorithm for T-MSS with  $\alpha$ -Lipschitz transformations has competitive ratio at least  $\min\{\alpha + 1, \alpha^2\}^{n-2}$ , even with randomization.*

The proof of the upper bound consists of three main steps: First we give a reduction to the case of 1-Lipschitz transformations in ultrametrics. Then we employ the fact that ultrametrics admit an ultrahomogeneous extension of size  $2^{n-1}$ , which will be proved later in Section 5.2. Finally, we show that the WFA achieves the optimal competitive ratio of  $n - 1$  for this special case of 1-Lipschitz transformations on ultrahomogeneous ultrametrics.

#### 3.1 From $\alpha$ -Lipschitz in general metrics to 1-Lipschitz in ultrametrics

For two metrics  $d$  and  $\hat{d}$  defined on a set  $M$ , we say that  $\hat{d}$  is an  $\alpha$ -distortion of  $d$ , for  $\alpha \geq 1$ , if for any  $x, y \in M$  we have  $d(x, y) \leq \hat{d}(x, y) \leq \alpha \cdot d(x, y)$ .

► **Lemma 7.** *Fix a constant  $\alpha \geq 2$  and an  $n$ -point metric space  $(M, d)$ . There is an ultrametric  $\hat{d}$  on  $M$  that is an  $(\alpha + 1)^{n-2}$ -distortion of  $d$ , and any transformation  $f: A \subseteq M \rightarrow M$  that is  $\alpha$ -Lipschitz with respect to  $d$  is 1-Lipschitz with respect to  $\hat{d}$ .*

**Proof.** The idea is to group the edges (i.e., pairs of distinct elements) in  $M$  into levels according to their distance. In level 1 we start with the minimum distance edges, and repeatedly add all edges which are within a factor  $\alpha$  of the largest level 1 edge until no more are possible. We then continue, constructing level  $k$  by starting with the shortest edge not in a previous level, and then adding all edges within a factor  $\alpha$  of some edge already in level  $k$ . Let  $L_k$  be the longest distance of an edge in level  $k$ . We define  $\hat{d}$  by setting level  $k$  edges to have  $\hat{d}$ -length  $L_k$ .

Note that since  $\alpha \geq 2$ , being connected by edges of level at most  $k$  is an equivalence relation for any  $k$ . Therefore, defining distances to be  $L_k$  in level  $k$  is a valid ultrametric. Since no edge has length in any interval  $(L_k, \alpha L_k]$ , any transformation that is  $\alpha$ -Lipschitz with respect to  $d$  is 1-Lipschitz with respect to  $\hat{d}$ .

It remains to argue that distances were increased by at most a factor  $(\alpha + 1)^{n-2}$  in going from  $d$  to  $\hat{d}$ . We show it for the edges in level  $k$ . Actually we may assume without loss of generality that  $k = 1$ , because if not we may take all edges in levels up to  $k - 1$  and set their distances to the shortest edge distance in level  $k$ . This is still a valid metric and the case of level 1 in this new metric implies the case of level  $k$  in the old metric. For the main argument when  $k = 1$ , consider adding the edges in order starting from an empty graph on the  $n$  points of  $M$ , giving an increasing sequence  $H_0, H_1, \dots$  of graphs. Exactly  $n - 1$  of these edges decrease the number of connected components by 1 at the time they are added. We call these edges critical and denote by  $D_1 \leq \dots \leq D_{n-1}$  their lengths. At any time, the critical edges in  $H_t$  form a spanning forest for  $H_t$ . Therefore the maximum distance of any edge in  $H_t$  is at most the sum of the lengths of the critical edges in  $H_t$ . Therefore the next critical edge to be added has length

$$D_s \leq \alpha \cdot \sum_{r=1}^{s-1} D_r.$$

From this it is easy to see inductively that  $D_s \leq \alpha(\alpha + 1)^{s-2}D_1$  for  $s \geq 2$ . This means the maximum distance of any edge in  $H$  is at most  $\sum_s D_s \leq (\alpha + 1)^{n-2}D_1$  (since there are  $n - 1$  critical edges). We remark that equality is achieved by the set of points  $\{0, 1, \alpha + 1, (\alpha + 1)^2, \dots, (\alpha + 1)^{n-2}\} \subset \mathbb{R}$ . ◀

### 3.2 WFA for 1-Lipschitz transformations on ultrahomogeneous ultrametrics

We now prove that WFA is  $(n - 1)$ -competitive for 1-Lipschitz transformations in ultrahomogeneous ultrametrics, thereby also proving the first statement of Theorem 5. Note that this bound is optimal, since  $n - 1$  is also the exact competitive ratio of ordinary MSS on any  $n$ -point metric space.

► **Lemma 8.** *Let  $(M, d)$  be an ultrahomogeneous ultrametric with  $n$  points. The WFA for T-MSS on  $(M, d)$  with 1-Lipschitz transformation is  $(n - 1)$ -competitive.*

**Proof.** We use the same potential function that also yields  $(n - 1)$ -competitiveness for ordinary MSS on general  $n$ -point metrics and  $(2n - 1)$ -competitiveness for MTS [7]:

$$\Phi(w) := \sum_{p \in M} w(p).$$

The bound (1) follows from the fact that  $w_t^-(p) - w_{t-1}(p) \geq 0$  for all  $p$ . Bound (3) for  $\rho_M = n$  follows from the 1-Lipschitzness of  $w$ , choosing  $C_M$  to be  $n - 1$  times the diameter of  $M$ .

For (2), we need to show that the sum of work function values is non-decreasing when a transformation  $f_t: A_t \rightarrow B_t$  is applied. On a high level, the idea is as follows. The work function  $w_t^-$  before transformation is supported on  $A_t$ , and the work function  $w_t$  afterwards on  $B_t$ . Imagine for simplicity that the work function value of all support points were 0. Then the work function value at other points is simply the distance from the support. Since  $f_t$  is 1-Lipschitz, we can think of  $B_t$  as a contracted version of the set  $A_t$  (and since the metric space is ultrahomogeneous, we can ignore the fact that  $B_t$  might be located in a very different part of the metric space than  $A_t$ ). This shrinking of the support means that most other points of the metric space tend to get further away from the support, and thus their work function values tend to increase.

We now turn to a formal proof of inequality (2). Since  $M$  is an ultrametric, we can view it as the set of leaves of an ultrametric tree.

Denote by  $T_r(p) := \{x \in M: d(p, x) \leq r\}$  the ball of radius  $r$  around  $p$ . Note that this is the set of leaves of the subtree rooted at the highest ancestor of  $p$  whose weight is at most  $r$ . In particular, the sets  $T_r(p)$  form a laminar family. We claim for all  $a, a' \in A_t$  and  $r, r' \geq 0$  that

$$T_r(f_t(a)) \cap T_{r'}(f_t(a')) = \emptyset \implies T_r(a) \cap T_{r'}(a') = \emptyset. \quad (5)$$

To see this, suppose  $x \in T_r(a) \cap T_{r'}(a')$  and say without loss of generality that  $r \leq r'$ . Then, by the ultrametric inequality,  $d(a, a') \leq \max\{d(a, x), d(a', x)\} \leq r'$ . Since  $f_t$  is 1-Lipschitz, this also means that  $d(f_t(a), f_t(a')) \leq r'$ . But then  $f_t(a) \in T_r(f_t(a)) \cap T_{r'}(f_t(a'))$ .

For  $y \geq 0$  and a work function  $w$  supported on a set  $S \in M$ , its  $y$ -sublevel set is given by

$$w^{-1}([0, y]) = \bigcup_{s \in S} T_{y-w(s)}(s).$$

Recall that  $w_t^-$  is supported on  $A_t$ , and note that  $w_t$  is supported on a set  $\tilde{B}_t \subseteq B_t$  such that

$$\forall b \in \tilde{B}_t \exists a_b \in A_t : f_t(a_b) = b \text{ and } w_t(b) = w_t^-(a_b).$$

Since the sets  $T_r(p)$  form a laminar family, we can choose for each  $y \geq 0$  a subset  $\tilde{B}_t^y \subseteq \tilde{B}_t$  such that

$$w_t^{-1}([0, y]) = \dot{\bigcup}_{b \in \tilde{B}_t^y} T_{y-w_t(b)}(b)$$

is a disjoint union. Due to implication (5), this also means that

$$\dot{\bigcup}_{b \in \tilde{B}_t^y} T_{y-w_t^-(a_b)}(a_b)$$

is a disjoint union.

Now, the cardinality of the  $y$ -sublevel set of  $w_t$  is bounded by

$$\begin{aligned} |w_t^{-1}([0, y])| &= \sum_{b \in \tilde{B}_t^y} |T_{y-w_t(b)}(b)| \\ &= \sum_{b \in \tilde{B}_t^y} |T_{y-w_t^-(a_b)}(a_b)| \end{aligned} \tag{6}$$

$$\begin{aligned} &= \left| \dot{\bigcup}_{b \in \tilde{B}_t^y} T_{y-w_t^-(a_b)}(a_b) \right| \\ &\leq \left| \bigcup_{a \in A_t} T_{y-w_t^-(a)}(a) \right| \\ &= |(w_t^-)^{-1}([0, y])|, \end{aligned} \tag{7}$$

where equation (6) uses the fact that since  $M$  is ultrahomogeneous, balls of the same radius have the same cardinality. Thus, for each  $y \geq 0$  there are at least as many points whose  $w_t^-$ -value is at most  $y$  as there are points whose  $w_t$ -value is at most  $y$ . Therefore, if  $p_1, p_2, \dots, p_n$  and  $p_1^-, p_2^-, \dots, p_n^-$  are two enumerations of  $M$  by increasing  $w_t$ - and  $w_t^-$ -values, respectively, then  $w_t^-(p_i^-) \leq w_t(p_i)$ . Hence, inequality (2) follows.  $\blacktriangleleft$

► **Remark.** One may wonder whether the guarantee of Lemma 8 is achieved more generally for ultrahomogeneous *metric* (rather than ultrametric) spaces. The answer is negative: Consider the 8-point metric space  $M = \{0, 2\} \times \{0, 3\} \times \{0, 4\}$  with the  $\ell_1$ -norm. Since  $M$  is isometric to the cube  $\{0, 1\}^3$  with a weighted  $\ell_1$ -norm, and all partial isometries on  $M$  are translations, Theorem 4 (proved in Section 5.3) implies that  $M$  is ultrahomogeneous. Consider the work function  $w$  supported  $\{(0, 0, 0), (2, 3, 0)\}$ , where it takes value 0. We have  $\Phi(w) = 2 + 2 + 4 + 4 + 6 + 6 = 24$ . The updated work function  $w'$  after contracting  $\{(0, 0, 0), (2, 3, 0)\}$  to  $\{(0, 0, 0), (0, 0, 4)\}$  has  $\Phi(w') = 2 \cdot (2 + 3 + 5) = 20 < \Phi(w)$ , meaning that inequality (2) is violated. Crucially, property (5) that disjoint balls remain disjoint when their centers are moved apart is violated on  $M$ . Using this observation, it is not hard to construct a request sequence on  $M$  where (1) is always tight for  $x = b_{t-1}$  and (2) is either tight or violated for each request, and violated for a constant fraction of the requests. For such a request sequence, the analysis in the proof of Lemma 6 yields a *lower* bound strictly larger than  $n - 1$  on the competitive ratio of WFA for T-MSS on  $M$  with 1-Lipschitz transformations.

### 3.3 Putting it together

Given any  $n$ -point metric  $(M, d)$ , we obtain a  $2 \max\{2\alpha + 2, 6\}^{n-2}$ -competitive deterministic algorithm for T-MSS with  $\alpha$ -Lipschitz transformations as follows: Making a multiplicative error of  $\max\{\alpha + 1, 3\}^{n-2}$ , Lemma 7 allows us to assume that  $M$  is an ultrametric and transformations are 1-Lipschitz. By Theorem 4, it further admits a  $2^{n-1}$ -point weakly ultrahomogeneous extension  $(\hat{M}, \hat{d})$ , and the proof of this statement in Theorem 4 actually shows that  $(\hat{M}, \hat{d})$  is still an ultrametric and it is ultrahomogeneous (not just weakly). Therefore, by Lemma 8, the WFA is  $2^{n-1}$ -competitive on  $(\hat{M}, \hat{d})$ . Overall, this gives a competitive ratio of  $\max\{\alpha + 1, 3\}^{n-2} 2^{n-1} = 2 \max\{2\alpha + 2, 6\}^{n-2}$ .

### 3.4 Lower bound for Lipschitz transformations

In this section we prove the lower bound part of Theorem 1 showing that any randomized algorithm for T-MSS with  $\alpha$ -Lipschitz transformations has competitive ratio at least  $\min\{\alpha + 1, \alpha^2\}^{n-2}$ .

Let  $m := \min\{\alpha + 1, \alpha^2\}$ . Assume  $\alpha \geq 1$  since otherwise there is nothing to show. Consider the graph with vertices  $p_1, \dots, p_n$  and edges from  $p_1$  to every vertex and between consecutive vertices of lengths

$$\begin{aligned} d(p_1, p_i) &:= m^{i-2} & i = 2, \dots, n \\ d(p_i, p_{i+1}) &:= \alpha m^{i-2} & i = 2, \dots, n-1. \end{aligned}$$

Note that these edge lengths satisfy the triangle inequality: When adding the vertices to the graph in order, the addition of  $p_i$  only creates the new triangle  $(p_1, p_{i-1}, p_i)$  with edge lengths  $(m^{i-3}, \alpha m^{i-3}, m^{i-2})$ . Since  $1 \leq \alpha \leq m \leq \alpha + 1$ , it satisfies the triangle inequality. Therefore, the shortest path extension of  $d$  defines a valid metric.

Consider a T-MSS instance on this space with  $p_1$  as its initial state. For  $t = 1, 2, \dots, n-2$ , we issue transformations

$$\begin{aligned} f_{2t-1} &: \{p_1, p_{t+1}\} \rightarrow \{p_{t+1}, p_{t+2}\} \\ f_{2t} &: \{p_{t+1}, p_{t+2}\} \rightarrow \{p_1, p_{t+2}\}, \end{aligned}$$

where each transformation maps the first (resp. second) point of the domain to the first (resp. second) point of the codomain. Note that these maps are  $\alpha$ -Lipschitz, using for  $f_{2t}$  that  $m \leq \alpha^2$ . Then, we issue the final transformation

$$f_{2n-3} : \{x\} \rightarrow \{p_1\}$$

with  $x$  chosen uniformly at random from  $\{p_1, p_n\}$ .

With probability  $1/2$ , the online algorithm has to pay  $m^{n-2}$  to move to  $x$  when  $f_{2n-3}$  is issued. An offline algorithm can serve the sequence with expected cost  $1/2$ : Before  $f_1$ , it either stays at  $p_1$  for cost 0 (if  $x = p_1$ ) or moves from  $p_1$  to  $p_2$  for cost 1 (if  $x = p_n$ ), which allows to serve the rest of the request sequence for free. Since the request sequence can be repeated arbitrarily often (with a new random  $x$ ), we conclude a lower bound of  $m^{n-2}$  on the competitive ratio.

## 4 Algorithms and Lower Bounds for Special Cases of T-MSS

In this section we show upper and lower bounds on the competitive ratio for T-MSS for special cases. In Section 4.1 we show that there exists no online algorithm with finite competitive ratio for nested convex body chasing with contractions in the plane, even with randomization.

In Section 4.2 we design a new randomized algorithm for the  $k$ -taxi problem. In Section 4.3 we show upper and lower bounds for swap transformations. Finally, in Section 4.4 we show a superlinear lower bound on the competitiveness of WFA for isometry transformations.

#### 4.1 Contracting convex bodies are unchaseable

Here we show that nested convex body chasing with contractions has no competitive algorithm. In nested convex body chasing, the requests form a nested sequence  $K_0 \supseteq K_1 \supseteq \dots$  of convex sets in  $\mathbb{R}^d$ . The player starts at  $x_0 \in K_0$  and moves online to a point  $x_t \in K_t$ , paying movement cost  $\sum_{t \geq 1} \|x_{t-1} - x_t\|$ . This problem is a special case of the more general convex body chasing problem which allows an arbitrary non-nested sequence  $K_0, K_1, \dots$  of convex sets. This problem has received a lot of recent study without transformations and admits a  $d$ -competitive algorithm – see [3, 1, 11, 10, 2, 20]. Because  $\mathbb{R}^d$  is ultrahomogenous, it follows that the  $d$ -competitive algorithm continues to apply with partial isometry transformations.

Here we show that with contraction transformations  $\mathbb{R}^d \rightarrow \mathbb{R}^d$  there is no competitive algorithm, even in the nested case for  $d = 2$  and with randomization. This gives a non-trivial example in which contractions are provably harder than isometries. In fact all the contractions we use are projections from  $K_t$  to  $K_{t+1}$ . Our proof goes by reduction to a family of 1-dimensional MTS problems known to have arbitrarily large competitive ratio. A related reduction appeared in [12] to show that 2-server convex body chasing is impossible in 2 dimensions.

► **Theorem 2.** *There exists no online algorithm with finite competitive ratio for nested convex body chasing with contractions in the plane, even with randomization.*

**Proof.** Fix a large integer  $n$ , a much larger integer  $M = M(n)$  and a much larger  $N = N(n)$ . We start with  $K_0 = [0, 1] \times [0, N]$  with starting point  $x_0 = (0, 0)$ . The projected convex sets rotate modulo 3: At multiples of 3 we simply have  $K_{3t} = [0, 1] \times [t, N]$ . Now, for a sequence  $(a_1, \dots, a_N)$  of positive integers  $a_i \in \{1, 2, \dots, n\}$  define the points  $p_1^t = (\frac{a_t}{n}, t)$ ,  $p_2^t = (\frac{a_t-1}{n}, t + \frac{1}{Mn})$  and  $p_3^t = (\frac{a_t+1}{n}, t + \frac{1}{Mn})$ . We define  $K_{3t+1}$  by cutting from  $K_{3t}$  along the lines  $p_1^t p_2^t$  and  $p_1^t p_3^t$ . We define  $K_{3t+2}$  by also cutting along the line  $p_2^t p_3^t$ . See Figure 1.

We take  $a_t$  to be an adversarially chosen sequence of such integers, yielding an adversarial nested chasing instance. For  $s$  congruent to 0 or 2 modulo 3, we apply projection maps (which are contractions) from  $K_s$  onto  $K_{s+1}$ . Hence movement cost is incurred only on transitions from  $K_{3t+1} \rightarrow K_{3t+2}$ .

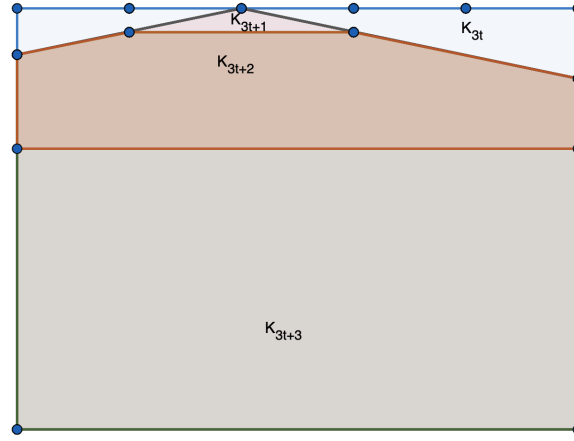
The idea is that the resulting problem is approximately a 1-dimensional MTS, as without loss of generality we may assume  $x_{3t}$  is always on the upper boundary of  $K_{3t}$ . It is easy to see that, crucially, the horizontal movement induced by the projections is  $O(M^{-2})$  per time-step – we will treat this as an additive error term. The vertical movement corresponds to a metrical task system with cost functions given by  $\frac{1}{M}c_{a_t}(x)$  for

$$c_{a_t}(x) = \begin{cases} 0 & \text{if } x \leq \frac{a_t-1}{n} \\ \min\{|x - \frac{a_t-1}{n}|, |x - \frac{a_t+1}{n}|\} & \text{if } x \in [\frac{a_t-1}{n}, \frac{a_t+1}{n}] \\ 0 & \text{if } x \geq \frac{a_t+1}{n}. \end{cases}$$

First, by repeating values of  $a_t$  in blocks of  $M$ , we can obtain an MTS which directly uses cost functions  $c_{a_t}(x)$ , now with an additive error  $O(M^{-1})$  per (block) time step for the horizontal movement. Next we claim any randomized algorithm for this new MTS can be assumed to stay on the finite set of values  $\frac{k}{n}$  for  $k \in \{0, 1, \dots, n\}$ . Indeed, if a randomized algorithm is at position  $\frac{k+\alpha}{n}$  for  $\alpha \in (0, 1)$  we can instead move to  $\frac{k}{n}$  with probability  $1 - \alpha$  and

$\frac{k+1}{n}$  with probability  $\alpha$ . Moreover we can couple these roundings together by first sampling  $u \in [0, 1]$  uniformly and rounding at all times based on whether  $u \leq \alpha$  or not. This turns any algorithm into a randomized algorithm which stays on the values  $\frac{k}{n}$  and has the same expected movement cost. Because the cost functions  $c_{a_t}$  are affine on each interval  $(\frac{k}{n}, \frac{k+1}{n})$  it results in the same expected service cost as well.

If the player is restricted to stay on the  $n + 1$  values  $\frac{k}{n}$ , the movement cost functions  $c_{a_t}$  are 0 at all but one of these values. Hence by repeating requests  $O_n(1)$  times to force movement we may reinterpret this as an  $n$ -server problem on a metric space with  $n + 1$  points, by taking the player's location in the original problem to be the unique spot with no server. It is well-known [5, 6] that the randomized competitive ratio of any  $n$ -server problem in a metric space with at least  $n + 1$  points is  $\Omega(\log n / \log \log n)$ . Finally, it is easy to see that for any MTS on a finite state space with competitive ratio  $C$ , an additive error in cost of  $o(1)$  per time-step affects the competitive ratio by  $o(1)$ . Therefore for any fixed  $n$ , taking  $M \rightarrow \infty$  results eventually in an MTS with competitive ratio  $\Omega(\log n / \log \log n)$  even taking the horizontal effects of the projection maps into account. Finally taking  $N$  sufficiently large to realize this competitive ratio gives the desired lower bound.  $\blacktriangleleft$



**Figure 1** To show that chasing nested convex bodies is impossible with contractions, we construct shrinking sets as shown. Euclidean nearest-point projections are taken onto the sets  $K_{3t+1}, K_{3t+3}$ , so that movement cost is incurred only in moving from  $K_{3t+1}$  to  $K_{3t+2}$ . Up to the negligible horizontal movements from projection, this results in a 1-dimensional metrical task system with unbounded competitive ratio.

## 4.2 A $\text{poly}(n, \log k)$ -competitive $k$ -taxi algorithm

In the  $k$ -taxi problem, there are  $k$  taxis located in a metric space  $(M, d)$ . A sequence of requests arrives, where each request is a pair of points  $(s_t, d_t) \in M \times M$ , representing the start and destination of a passenger request. Each request must be served upon its arrival by sending a taxi to  $s_t$ , from where it is relocated to  $d_t$ . The cost is defined as the distance travelled by taxis *while not carrying a passenger*, i.e., excluding the distances from  $s_t$  to  $d_t$ .

Note that the  $k$ -taxi problem is a special case of T-MSS: As the metric space for T-MSS, we take the set of taxi configurations (i.e.,  $k$ -point multisets of points in  $M$ ), and the distance between two configurations is the minimum cost of moving from one configuration to the other. A taxi request  $(s_t, d_t)$  translates to the transformation that maps configurations containing  $s_t$  to the corresponding configurations with a taxi at  $s_t$  replaced by a taxi at  $d_t$ . Observe that this transformation is an isometry in the configuration space.

In principle, the size of a weakly ultrahomogeneous extension of the configuration space would yield a bound on the competitive ratio of the  $k$ -taxi problem. However, we do not know a bound on this blow-up in general. We overcome this obstacle as follows: First, we apply a well-known embedding of the original  $k$ -taxi metric space into a tree (HST) metric. Then, we consider the configuration space of this tree metric, whose metric is given by a weighted  $\ell_1$ -norm. Moreover, the isometries corresponding to the  $k$ -taxi requests in this tree metric are actually translations, and for this case, Theorem 4 yields a bound on the blow-up (proved later in Section 5.3). The resulting algorithm has competitive ratio  $O((n \log k)^2 \log n)$ , improving upon the previous bound of  $O(2^k \log n)$  [14] whenever  $n$  is sub-exponential in  $k$ .

► **Theorem 3.** *There is a randomized  $O((n \log k)^2 \log n)$ -competitive algorithm for the  $k$ -taxi problem on  $n$ -point metrics.*

**Proof.** By well-known techniques [4, 16], any  $n$ -point metric space can be embedded with distortion  $O(\log n)$  into the set of leaves of a random (weighted) tree. It therefore suffices to describe an  $O((n \log k)^2)$ -competitive algorithm for the  $k$ -taxi problem on the set  $\mathcal{L}$  of leaves of a tree with  $|\mathcal{L}| = n$ . Notice that there is a tree  $\mathcal{T}$  with only  $O(n)$  vertices that induces the metric on  $\mathcal{L}$ .

Let  $V$  be the set of vertices of  $\mathcal{T}$  excluding the root. For  $v \in V$ , let  $w_v$  be the length of the edge from  $v$  to its parent. If we denote by  $x_v$  the number of taxis in the subtree rooted at  $v$ , then a configuration of  $k$  taxis can be denoted by a point in  $M := \{0, \dots, k\}^V$ . Notice that only some points in  $M$  correspond to valid  $k$ -taxi configurations. The cost of moving from configuration  $x$  to configuration  $y$  is given by the metric

$$d(x, y) := \sum_{v \in V} w_v |x_v - y_v|.$$

Thus, the  $k$ -taxi problem on  $\mathcal{T}$  is a special case of T-MSS on  $(M, d)$ . A  $k$ -taxi request  $(s_t, d_t)$  corresponds to a translation by the vector that has 1-entries in coordinates of ancestors of  $d_t$  that are not ancestors of  $s_t$ , -1-entries in coordinates of ancestors  $s_t$  that are not ancestors of  $d_t$ , and 0 in the remaining coordinates. We therefore need to extend  $M$  to a space  $\hat{M}$  where these translations extend to automorphisms. As stated in Theorem 4 and proved later in Section 5.3, such an extension  $\hat{M}$  of size  $(2k)^{|V|}$  exists. Thus, by running an  $O(\log^2 |\hat{M}|)$ -competitive algorithm for MSS on  $\hat{M}$  and treating each automorphism as a renaming of the points of  $\hat{M}$ , we obtain an algorithm for the  $k$ -taxi problem on  $\mathcal{T}$  with competitive ratio  $O(\log^2 |\hat{M}|) = O((n \log k)^2)$ , where we used that  $|V| = O(n)$ . Combined with the  $O(\log n)$  loss due to the tree embedding, the theorem follows. ◀

### 4.3 Competitiveness for swap transformations

In this section, we show tight bounds of  $2n - 3$  on the competitive ratio for T-MSS in general metrics when each transformation is either the identity on its domain (recall that we always allow identity transformations) or a swap, thereby proving the second part of Theorem 5. The upper bound is achieved by the WFA.

**Upper bound.** For brevity, we will denote the distance between two points  $x, y \in M$  by  $xy$  instead of  $d(x, y)$ . For  $X \subseteq M$  and  $x \in M$ , we write  $X - x := X \setminus \{x\}$ . For a work function  $w$  and  $x, y \in M$ , let



$$\begin{aligned}\Psi_{x,y}(w) &:= \text{cl}(M - x - y) + \sum_{p \in M - x - y} \min\{w(x) + yp, w(y) + xp\} \\ \Phi(w) &:= \sum_{p \in M} w(p) + \min_{x,y \in M} \Psi_{x,y}(w)\end{aligned}$$

Here,  $\text{cl}(X) := \sum_{\{x,y\} \subseteq X} xy$  denotes the size of the clique of  $X$ , i.e., the sum of all distances between points in  $X$ . It suffices to show that  $\Phi$  satisfies the properties of Lemma 6 with  $\rho_M = 2n - 2$ .

Inequality (3) is immediate from the fact that  $\Phi(w)$  is a sum of  $2n - 2$  function values of  $w$  and a bounded number of distances of  $M$ , and each function value of  $w$  differs from  $\min_x w(x)$  by at most the diameter of  $M$  due to the 1-Lipschitzness of  $w$ .

Inequality (1) follows from the fact that  $\Phi(w)$  contains the summand  $w(x)$ , the remaining summands of  $\Phi(w)$  are non-decreasing in  $w$ , and  $w_{t-1} \leq w_t^-$  pointwise.

Inequality (2) is trivial if  $f_t$  is the identity on its domain, so it remains to consider the case that  $f_t: \{a, b\} \rightarrow \{a, b\}$  is a swap. Both  $w_t^-$  and  $w_t$  are supported on  $\{a, b\}$ . We first show that if  $w$  is supported on  $\{a, b\}$ , then  $\Psi_{x,y}(w)$  is minimized when  $\{x, y\} = \{a, b\}$ .

Let  $x$  and  $y$  be such that  $\Psi_{x,y}(w)$  is minimized and suppose  $x \notin \{a, b\}$ . Then we can assume without loss of generality (by symmetry) that  $w(x) = w(a) + ax$ . Then  $\min\{w(x) + ya, w(y) + xa\} = w(y) + xa$ . Thus,

$$\begin{aligned}\Psi_{x,y}(w) &= \text{cl}(M - x - y) + w(y) + xa + \sum_{p \in M - x - y - a} \min\{w(a) + ax + yp, w(y) + xp\} \\ &\geq \text{cl}(M - x - y) + w(y) + xa \\ &\quad + \sum_{p \in M - x - y - a} (xp - ap + \min\{w(a) + yp, w(y) + ap\}) \\ &\geq \text{cl}(M - a - y) + \sum_{p \in M - y - a} \min\{w(a) + yp, w(y) + ap\} \\ &= \Psi_{a,y}(w).\end{aligned}$$

Thus,  $\Psi_{x,y}(w)$  is also minimized when  $x = a$ . If  $y \neq b$  and  $w(y) = w(b) + by$ , then the symmetric argument shows that  $\Psi_{x,y}(w)$  is minimized when  $\{x, y\} = \{a, b\}$ . Otherwise, if  $y \neq b$ , then  $w(y) = w(a) + ay$  since  $w$  is supported on  $\{a, b\}$ . Then

$$\begin{aligned}\Psi_{a,y}(w) &= \text{cl}(M - a - y) + \sum_{p \in M - a - y} \min\{w(a) + yp, w(a) + ay + ap\} \\ &= \text{cl}(M - a) + (n - 2)w(a) \\ &\geq \text{cl}(M - a - b) + \sum_{p \in M - a - b} \min\{w(a) + bp, w(b) + ap\} \\ &= \Psi_{a,b}(w).\end{aligned}$$

Thus, it is indeed the case that  $\min_{x,y} \Psi_{x,y}(w) = \Psi_{a,b}(w)$ , for both  $w = w_t^-$  and  $w = w_t$ . Hence,

$$\begin{aligned}\Phi(w_t^-) - \Phi(w_t) &= w_t^-(a) + w_t^-(b) - w_t(a) - w_t(b) \\ &\quad + \sum_{p \in M - a - b} (w_t^-(p) + \min\{w_t^-(a) + bp, w_t^-(b) + ap\}) \\ &\quad - \sum_{p \in M - a - b} (w_t(p) + \min\{w_t(a) + bp, w_t(b) + ap\}).\end{aligned}$$

Since  $w_t(a) = w_t^-(b)$ ,  $w_t(b) = w_t^-(a)$ , and  $w(p) = \min\{w(a) + ap, w(b) + bp\}$  for  $w = w_t^-$  and  $w = w_t$ , everything cancels in the last sum and we obtain (2).

**Lower bound.** Consider the metric with points  $1, 2, \dots, n$  where the distance from 1 to any other point is 1 and the distance between any other two points is 2. The initial location of the server is 1. For  $i = 1, \dots, n-1$ , the  $i$ th request is the identity with domain  $A_i$ , where  $A_1 := \{2, 3, \dots, n\}$  and for  $i \geq 2$ ,  $A_i$  is the subset of  $A_{i-1}$  obtained by removing the location of the online algorithm's server before this request is issued. During these requests, the online algorithm suffers cost  $2n-3$ , but an offline algorithm could immediately go to the one point  $p$  in  $A_{n-1}$  for cost 1. We issue one more request that swaps  $p$  and 1 so as to return to the initial configuration, allowing to repeat the procedure arbitrarily often.

#### 4.4 Superlinear lower bound for WFA with isometries

In this section we show a superlinear lower bound of  $\omega(n^{1.29})$  on the competitiveness of WFA for T-MSS with isometry transformations, proving the third part of Theorem 5.

It suffices to show the statement for values of  $n$  that are a power of 4. Let  $\alpha \in \mathbb{N}$  be some large constant. For  $h \in \mathbb{N}_0$ , we construct a  $4^h$ -point metric space  $T_h$  by induction: The space  $T_0$  is just a single point. For  $h \geq 1$ , space  $T_h$  is a disjoint union of four copies of  $T_{h-1}$ , which we denote  $T_{h-1}^0, T_{h-1}^1, T_{h-1}^2, T_{h-1}^3$ . For points  $x, y$  from two different copies of  $T_{h-1}$  we define their distance as  $\alpha^h$  if one of the copies is  $T_{h-1}^0$  and as  $2\alpha^h$  otherwise.

For a set  $X \subseteq T_{h-1}$  and  $i = 0, 1, 2, 3$ , denote by  $X^i$  the copy of  $X$  in  $T_{h-1}^i$ , and similarly if  $X$  is a point rather than a set. We define a special point  $s_h \in T_h$  as follows:  $s_0$  is the single point in  $T_0$ . For  $h \geq 1$ ,  $s_h := s_{h-1}^0$ .

We consider T-MSS on  $T_h$  when the server starts at  $s_h$ . Let  $w_0 = d(\cdot, s_h)$  be the initial work function. We will construct a request sequence  $\sigma_h$  during which WFA suffers cost  $(6^h - 1)\alpha^h(1 - o(1))$  as  $\alpha \rightarrow \infty$  and at whose end the work function is at most  $\alpha^h + w_0$  pointwise, with WFA returning to  $s_h$  in the end. Since such a request sequence can be repeated, it will imply that the competitive ratio is at least  $6^h - 1 = n^{(\ln 6)/(\ln 4)} - 1 = \omega(n^{1.29})$ .

For  $h = 0$ , we simply choose the empty request sequence. Consider now  $h \geq 1$ . For a partial isometry  $f: A \rightarrow B$  of  $T_{h-1}$  and  $i = 0, 1, 2, 3$ , denote by  $f^i: A^i \cup \bigcup_{j \neq i} T_{h-1}^j \rightarrow B^i \cup \bigcup_{j \neq i} T_{h-1}^j$  the map that acts like  $f$  on  $A^i$  and is the identity on  $T_{h-1}^j$  for  $j \neq i$ . Note that  $f^i$  is a partial isometry of  $T_h$ . Denote by  $\sigma_{h-1}^i$  the sequence obtained by extending each partial isometry in  $\sigma_{h-1}$  (the sequence from the induction hypothesis) in this way.

We construct  $\sigma_h$  as follows: First, we issue  $2\alpha - 2$  copies of  $\sigma_{h-1}^0$ . Since each work function  $w$  during this sequence admits a point  $p \in T_{h-1}^0$  with  $w(p) \leq (2\alpha - 2)\alpha^{h-1}$ , but  $w(x) = \alpha^h$  for all  $x \in \bigcup_{j \neq 0} T_{h-1}^j$ , WFA will stay within  $T_{h-1}^0$  during these requests, suffering cost  $(6^{h-1} - 1)2\alpha^h(1 - o(1))$ . Then we issue the identity request with domain  $\{s_{h-1}^1, s_{h-1}^2, s_{h-1}^3\}$ , forcing the algorithm to move to one of these three points for cost  $\alpha^h$ . By symmetry, we can assume without loss of generality that WFA moves to  $s_{h-1}^1$ . We now issue  $2\alpha - 2$  copies of  $\sigma_{h-1}^1$ , followed by the identity request with domain  $\{s_{h-1}^2, s_{h-1}^3\}$ . Similarly to before, WFA again suffers cost  $(6^{h-1} - 1)2\alpha^h(1 - o(1))$  and then moves to, say,  $s_{h-1}^2$  for cost  $2\alpha^h$ . Finally, we issue  $2\alpha - 2$  copies of  $\sigma_{h-1}^2$  followed by the request  $\{s_{h-1}^3\} \rightarrow \{s_h\}$ ,  $s_{h-1}^3 \mapsto s_h$ , increasing WFA's cost by another  $(6^{h-1} - 1)2\alpha^h(1 - o(1)) + 2\alpha^h$ . Overall, WFA suffers cost

$$(6^{h-1} - 1)(2 + 2 + 2)\alpha^h(1 - o(1)) + (1 + 2 + 2)\alpha^h = (6^h - 1)\alpha^h(1 - o(1)),$$

as claimed. Moreover, the final work function is  $\alpha^h + w_0$  because an offline algorithm could move to  $s_{h-1}^3$  for cost  $\alpha^h$  at the start of the request sequence, suffer no cost during the rest of the sequence, and be mapped back to  $s_h$  (for free) via the final request.

## 5 Bounds on the Metric Extension Blow-up

In this section we prove upper and lower bounds on the blow-up for several families of metric spaces and transformations, proving Theorem 4.

► **Theorem 4.** *The following bounds on the blow-up are tight.*

Family of metric spaces	Family of isometries	Blow-up
Ultrametrics	General	$2^{n-1}$
Ultrametrics: $k$ distinct non-zero distances	General	$\approx \left(\frac{n+k-1}{k}\right)^k$ <sup>4</sup>
Equally spaced points on a line	General	$2n - 2$
$(\{0, \dots, k\}^D, \text{weighted } \ell_1)$	Translations	$(2k)^D$
General metrics	Swaps	$2^{n-1}$

It was shown independently by Solecki [21] and Vershik [22] that every *finite* metric space  $M$  admits a *finite*<sup>5</sup> weakly ultrahomogeneous extension  $\hat{M}$ . An elementary proof of this result was presented very recently in [18]. However, the main part of the construction in [18] consists of  $\lceil R \rceil$  growing steps, where  $R$  is the aspect ratio of  $M$ , and a naive bound on the growth factor in the  $i$ th step alone is already doubly exponential in  $i$ . Thus, this does not yield an upper bound on the cardinality of  $\hat{M}$  in terms of the cardinality of  $M$ , but only one that also involves the aspect ratio. Thus, even though there exists a finite extension for any  $n$ -point metric, it is unclear whether its size can be bounded as a function of  $n$ . If not, this would mean that the blow-up for general metrics and isometries is infinite.

### 5.1 General metrics with swap transformations

Let  $M$  be an  $n$ -point metric. We will show how to extend  $M$  to a  $2^{n-1}$ -point space where every swap extends to an automorphism. The tightness of this upper bound follows from the lower bound for ultrametrics (with  $n - 1$  distinct distances) proved in Section 5.2. There, we will show that even if only maps with 1-point domain need to extend to automorphisms, the extended space may require cardinality  $2^{n-1}$ .

We embed  $M$  into the vector space  $\hat{M} = \mathbb{F}_2^{n-1}$  by enumerating the points  $p_1, \dots, p_n$  of  $M$  in arbitrary order and defining the embedding  $\varphi : p_k \mapsto (1^{k-1}, 0^{n-k})$ . We choose the metric on  $\hat{M}$  to extend that of  $M$  and also be translation invariant, and explain just below why such a choice exists. Now for  $x, y \in \mathbb{F}_2^{n-1}$ , their swap is translation by  $(x + y)$ . Since the metric on  $\hat{M}$  is translation invariant, this translation gives the desired extension to an automorphism on  $\hat{M}$ .

Now we explain why there exists such a translation invariant metric on  $\hat{M}$ . We first extend to a partial metric  $(\hat{M}, \hat{d})$  only by translation invariance, i.e.,  $\hat{d}$  is the partial function on  $\hat{M} \times \hat{M}$  defined by  $\hat{d}(x, y) = d(p_i, p_j)$  if  $x - y = \varphi(p_i) - \varphi(p_j)$ . Viewing  $\hat{d}$  as a weighted graph  $G$ , the shortest path extension of  $\hat{d}$  is also translation invariant. Moreover, it gives a valid metric on  $\hat{M}$  if and only if there is no cycle in  $G$  violating the triangle inequality.

Now, the values  $\varphi(p_i) - \varphi(p_j) \in \hat{M}$  range over vectors with a single continuous block of 1s. A cycle in  $G$  consists of a multiset  $\mathcal{S}$  of such vectors adding to 0. Viewing  $\varphi(p_i) - \varphi(p_j)$  as an edge connecting  $p_i$  and  $p_j$ , we claim that any such multiset  $\mathcal{S}$  must correspond to a

<sup>4</sup> The precise blow-up is  $(a + 1)^b a^{k-b}$  where  $a = \lfloor \frac{n+k-1}{k} \rfloor$  and  $b = (n - 1) \bmod k$ .

<sup>5</sup> The existence of an *infinite* ultrahomogeneous metric space that extends every finite metric space, called the Urysohn universal space, has been known since the 1920s.

multigraph on  $M$  with even degree at each vertex  $p_i$ . Indeed, for  $i \geq 2$ , the parity of the degree at  $p_i$  is the difference between the coordinates  $i-1$  and  $i$  in the summed vector, which is 0 by definition (treat coordinate  $n$  as being identically 0). Since the sum of degrees is even, also  $p_1$  must have even degree. Therefore this multigraph has an Eulerian circuit and in particular a cycle containing each edge. Since the edge weights are now exactly distances in  $M$ , we are done by applying the triangle inequality in  $M$ .

## 5.2 Ultrametrics

We consider ultrametrics with at most  $k$  distinct non-zero distances. Note that any  $n$ -point ultrametric has at most  $n-1$  distinct distances, so the general bounds on ultrametrics follow from the case  $k = n-1$ .

**Upper bound.** Recall that ultrametric spaces may be viewed as the leaves of a rooted tree in which vertices with lowest common ancestor at level  $i$  have distance  $L_i$ , where  $0 < L_1 < L_2 < \dots < L_k$  are the possible distances.

We construct  $\hat{M}$  by augmenting the  $n$ -leaf tree corresponding to  $M$  with additional points to create a symmetric tree  $\hat{M}$  where partial isometries extend to automorphisms. We claim this can be done with at most  $\left(\frac{n+k-1}{k}\right)^k$  total leaves. Indeed, the original tree's non-leaf vertices have  $C_1, \dots, C_j$  children for some numbers satisfying  $\sum_{i=1}^j (C_i - 1) = n - 1$ . Therefore letting  $E_i$  be the maximal number of children for any vertex at level  $i$ , we have  $\sum_i (E_i - 1) \leq n - 1$ . We take  $\hat{M}$  to be the leaves of a fully symmetric tree in which every vertex at level  $i$  has  $E_i$  children. It is clear that any partial isometry of this symmetric tree extends to an automorphism. Moreover  $|\hat{M}| = \prod_i E_i$ . Given the constraint  $\sum_{i=1}^k (E_i - 1) \leq n - 1$ , the bound  $|\hat{M}| \leq \left(\frac{n+k-1}{k}\right)^k$  follows from AM-GM. Since each  $E_i$  is an integer, the precise bound is  $(a+1)^b a^{k-b}$  if  $n-1 = ak + b$  for  $b \in \{0, 1, \dots, k-1\}$ .

**Lower bound.** Let  $M_0, \dots, M_k$  be disjoint sets, where  $M_0 = \{p_0\}$  is a singleton and  $M_1, \dots, M_k$  have cardinalities  $\lfloor \frac{n-1}{k} \rfloor$  or  $\lceil \frac{n-1}{k} \rceil$  such that the union  $M := M_0 \dot{\cup} M_1 \dot{\cup} \dots \dot{\cup} M_k$  has cardinality  $n$ . We define an ultrametric on  $M$  by defining the distance between any two distinct points  $x \in M_i, y \in M_j$  with  $i \leq j$  to be  $2 \cdot 3^{j-1}$ .

Let  $\hat{M} \supseteq M$  be a weakly ultrahomogeneous extension of  $M$ . For a point  $x \in \hat{M}$  and  $j = 0, \dots, k$ , denote by  $B_j(x)$  the set of points in  $\hat{M}$  within distance strictly less than  $3^j$  from  $x$ . We claim that  $|B_j(p_0)| \geq \prod_{i=1}^j (|M_i| + 1)$  for each  $j = 0, \dots, k$ . This implies that  $|\hat{M}| \geq \prod_{i=1}^k (|M_i| + 1) \geq \prod_{i=1}^k \lfloor \frac{n+k-1}{k} \rfloor$ .

To prove the claim, we proceed by induction on  $j$ . Clearly it is true for  $j = 0$ . For  $j \geq 1$ , consider the balls  $B_{j-1}(p)$  for  $p \in M_j \cup \{p_0\}$ . By the triangle inequality, they are disjoint and contained in  $B_j(p_0)$ . Since for each  $p \in M_j$  the map  $p_0 \mapsto p$  extends to an isomorphism, they must also all have the same cardinality, which is at least  $\prod_{i=1}^{j-1} (|M_i| + 1)$  by the induction hypothesis. Thus,  $B_j(p_0)$  has cardinality at least  $\prod_{i=1}^j (|M_i| + 1)$ .

## 5.3 The line and weighted $\ell_1$ -norms with translations

The extension of  $n$  equally spaced points on a line is simple: We extend it to a circle of  $2n-2$  equally spaced points. It is easy to see that the circle is ultrahomogeneous because any (partial) isometry is a combination of a rotation and possibly a reflexion. We will now extend this idea to multiple dimensions.

Consider the space  $M := \{0, 1, \dots, k\}^D$  with the distance given by the weighted  $\ell_1$ -norm  $d(x, y) := \sum_{i=1}^D w_i |x_i - y_i|$ , where  $w_1, \dots, w_D$  are arbitrary positive weights. As partial isometries, we consider the family of translations  $x \rightarrow x + v$  that map a subset of  $M$  to another subset of  $M$ . We will show that the associated blow-up is precisely  $(2k)^D$ . Note that the lower bound for  $D = 1$  also yields a tight lower bound of  $2n - 2$  for the blow-up of equally spaced points on a line,

**Upper bound.** Notice that any translation is a composition of translations of the form  $x \rightarrow x + e_i$  and their inverses, where  $e_i \in \{0, 1\}^D$  is the vector with a 1-entry in only the  $i$ th coordinate. It therefore suffices to extend  $M$  to a metric space  $\hat{M}$  where partial isometries of this restricted type extend to global isometries.

We extend  $M$  to the space  $\hat{M} = \{0, \dots, 2k - 1\}^D$  and define a metric on  $\hat{M}$  by

$$\hat{d}(x, y) := \sum_{i=1}^D w_i \min\{|x_i - y_i|, 2k - |x_i - y_i|\}.$$

This is the metric induced by the weighted  $\ell_1$ -norm when viewing  $\hat{M}$  as a  $D$ -dimensional torus. Clearly,  $\hat{d}$  extends  $d$ . Moreover, any isometry  $x \rightarrow x + e_i$  defined on a subset of  $M$  extends to the automorphism  $x \rightarrow x + e_i \bmod 2k$  on  $\hat{M}$ , where the “mod  $2k$ ” is applied coordinate-wise.

**Lower bound.** Let  $A_0 := \{0, 1, \dots, k\}$  and  $A_1 := \{0, 1\}$ . For a 0-1-string  $i_1 i_2 \dots i_D$ , consider the translation

$$\begin{aligned} f_{i_1 \dots i_D} : A_{i_1} \times \dots \times A_{i_D} &\rightarrow M \\ x &\mapsto x + (k - 1) \cdot (i_1, \dots, i_D). \end{aligned}$$

The choice of domain of  $f_{i_1 \dots i_D}$  is just to ensure that the image is still in  $M$ .

Let  $\hat{M} \supseteq M$  be an extension of  $M$  such that each  $f_{i_1 \dots i_D}$  extends to an automorphism  $\hat{f}_{i_1 \dots i_D}$  of  $\hat{M}$ .

Let  $C_0 := \{0, 1, \dots, k - 1\}$ ,  $C_1 := \{1, 2, \dots, k\}$  and  $S_{i_1 \dots i_D} := \hat{f}_{i_1 \dots i_D}(C_{i_1} \times \dots \times C_{i_D})$ .

Note that each set  $S_{i_1 \dots i_D}$  has cardinality  $k^D$ , and there are  $2^D$  such sets in total, corresponding to the  $2^D$  possible 0-1-strings of length  $D$ . Thus, the lower bound of  $(2k)^D$  on the cardinality of  $\hat{M}$  follows from the following claim.

▷ **Claim 9.** The sets  $S_{i_1 \dots i_D}$  are pairwise disjoint for different 0-1-strings  $i_1 \dots i_D$ .

**Proof.** Let  $y \in S_{i_1 \dots i_D}$  for some 0-1-string  $i_1 \dots i_D$ . We will show that  $i_1 \dots i_D$  is uniquely determined by  $y$ .

We can write  $y = \hat{f}_{i_1 \dots i_D}(x)$  for some  $x \in M$ . It suffices to show that  $i_j = 0$  if and only if  $y$  is closer to  $k \cdot \mathbb{1} - e_j$  than to  $k \cdot \mathbb{1}$ , where  $\mathbb{1}$  denote the all-ones vector. Equivalently, we will show that

$$i_j = 0 \iff d(x, f_{i_1 \dots i_D}^{-1}(k \cdot \mathbb{1} - e_j)) < d(x, f_{i_1 \dots i_D}^{-1}(k \cdot \mathbb{1})).$$

Note that the preimages  $f_{i_1 \dots i_D}^{-1}(k \cdot \mathbb{1})$  and  $f_{i_1 \dots i_D}^{-1}(k \cdot \mathbb{1} - e_j)$  exist in  $M$ , and they differ only in their  $j$ th entry.

If  $i_j = 0$ , then  $x_j \leq k - 1$  (by definition of  $C_{i_j}$ ) and the  $j$ th entries of  $f_{i_1 \dots i_D}^{-1}(k \cdot \mathbb{1} - e_j)$  and  $f_{i_1 \dots i_D}^{-1}(k \cdot \mathbb{1})$  are  $k - 1$  and  $k$ , respectively. Thus,  $x$  is closer to  $f_{i_1 \dots i_D}^{-1}(k \cdot \mathbb{1} - e_j)$  than to  $f_{i_1 \dots i_D}^{-1}(k \cdot \mathbb{1})$ .

If  $i_j = 1$ , then  $x_j \geq 1$  (by definition of  $C_{i_j}$ ) and the  $j$ th entries of  $f_{i_1 \dots i_D}^{-1}(k \cdot \mathbb{1} - e_j)$  and  $f_{i_1 \dots i_D}^{-1}(k \cdot \mathbb{1})$  are 0 and 1, respectively. Thus,  $x$  is further from  $f_{i_1 \dots i_D}^{-1}(k \cdot \mathbb{1} - e_j)$  than from  $f_{i_1 \dots i_D}^{-1}(k \cdot \mathbb{1})$ . ◁

## References

- 1 CJ Argue, Sébastien Bubeck, Michael B Cohen, Anupam Gupta, and Yin Tat Lee. A nearly-linear bound for chasing nested convex bodies. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 117–122. SIAM, 2019.
- 2 CJ Argue, Anupam Gupta, Guru Guruganesh, and Ziyue Tang. Chasing convex bodies with linear competitive ratio. In *Proceedings of the 31st Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1519–1524. SIAM, 2020.
- 3 Nikhil Bansal, Martin Böhm, Marek Eliáš, Grigorios Koumoutsos, and Seeun William Umboh. Nested convex bodies are chaseable. *Algorithmica*, pages 1–14, 2019.
- 4 Yair Bartal. Probabilistic approximations of metric spaces and its algorithmic applications. In *37th Annual Symposium on Foundations of Computer Science, FOCS '96*, pages 184–193, 1996.
- 5 Yair Bartal, Béla Bollobás, and Manor Mendel. Ramsey-type theorems for metric spaces with applications to online problems. *J. Comput. Syst. Sci.*, 72(5):890–921, 2006.
- 6 Yair Bartal, Nathan Linial, Manor Mendel, and Assaf Naor. On metric Ramsey-type phenomena. *Ann. of Math. (2)*, 162(2):643–709, 2005.
- 7 Allan Borodin and Ran El-Yaniv. *Online computation and competitive analysis*. Cambridge University Press, 1998.
- 8 Allan Borodin, Nathan Linial, and Michael E. Saks. An optimal on-line algorithm for metrical task system. *Journal of the ACM*, 39(4):745–763, 1992.
- 9 Sébastien Bubeck, Michael B. Cohen, James R. Lee, and Yin Tat Lee. Metrical task systems on trees via mirror descent and unfair gluing. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019*, pages 89–97, 2019.
- 10 Sébastien Bubeck, Bo'az Klartag, Yin Tat Lee, Yuanzhi Li, and Mark Sellke. Chasing nested convex bodies nearly optimally. In *Proceedings of the 31st Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1496–1508. SIAM, 2020.
- 11 Sébastien Bubeck, Yin Tat Lee, Yuanzhi Li, and Mark Sellke. Competitively chasing convex bodies. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 861–868, 2019.
- 12 Sébastien Bubeck, Yuval Rabani, and Mark Sellke. Online multiserver convex chasing and optimization. *arXiv preprint*, 2020. [arXiv:2004.07346](https://arxiv.org/abs/2004.07346).
- 13 Marek Chrobak and Lawrence L. Larmore. The server problem and on-line games. In *On-Line Algorithms, Proceedings of a DIMACS Workshop*, pages 11–64, 1991. doi:10.1090/dimacs/007/02.
- 14 Christian Coester and Elias Koutsoupias. The online  $k$ -taxi problem. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019*, pages 1136–1147. ACM, 2019.
- 15 Christian Coester and James R. Lee. Pure entropic regularization for metrical task systems. In Alina Beygelzimer and Daniel Hsu, editors, *Conference on Learning Theory, COLT 2019, 25-28 June 2019, Phoenix, AZ, USA*, volume 99 of *Proceedings of Machine Learning Research*, pages 835–848, 2019.
- 16 Jittat Fakcharoenphol, Satish Rao, and Kunal Talwar. A tight bound on approximating arbitrary metrics by tree metrics. *J. Comput. Syst. Sci.*, 69(3):485–497, 2004.
- 17 Amos Fiat, Yuval Rabani, and Yiftach Ravid. Competitive  $k$ -server algorithms (extended abstract). In *31st Annual Symposium on Foundations of Computer Science, FOCS '90*, pages 454–463, 1990. doi:10.1109/FSCS.1990.89566.
- 18 Jan Hubička, Matěj Konečný, and Jaroslav Nešetřil. A combinatorial proof of the extension property for partial isometries. *arXiv preprint*, 2018. [arXiv:1807.10976](https://arxiv.org/abs/1807.10976).
- 19 Mark Manasse, Lyle McGeoch, and Daniel Sleator. Competitive algorithms for on-line problems. In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 322–333. ACM, 1988.

- 20   Mark Sellke. Chasing convex bodies optimally. In *Proceedings of the 31st Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1509–1518. SIAM, 2020.
- 21   Sławomir Solecki. Extending partial isometries. *Israel Journal of Mathematics*, 150(1):315–331, 2005.
- 22   Anatoly M. Vershik. Globalization of the partial isometries of metric spaces and local approximation of the group of isometries of urysohn space. *Topology and its Applications*, 155(14):1618–1626, 2008.
- 23   Hsien-Chung Wang. Two-point homogeneous spaces. *Annals of Mathematics*, 55(1):177–191, 1952.



# Tight Hardness Results for Training Depth-2 ReLU Networks\*

**Surbhi Goel**

Microsoft Research, New York, NY, USA  
goel.surbhi@microsoft.com

**Adam Klivans**

The University of Texas at Austin, TX, USA  
klivans@cs.utexas.edu

**Pasin Manurangsi**

Google Research, Mountain View, CA, USA  
pasin@google.com

**Daniel Reichman**

Worcester Polytechnic Institute, MA, USA  
dreichman@wpi.edu

---

## Abstract

We prove several hardness results for training depth-2 neural networks with the ReLU activation function; these networks are simply weighted sums (that may include negative coefficients) of ReLUs. Our goal is to output a depth-2 neural network that minimizes the square loss with respect to a given training set. We prove that this problem is NP-hard already for a network with a single ReLU. We also prove NP-hardness for outputting a weighted sum of  $k$  ReLUs minimizing the squared error (for  $k > 1$ ) even in the realizable setting (i.e., when the labels are consistent with an unknown depth-2 ReLU network). We are also able to obtain lower bounds on the running time in terms of the desired additive error  $\epsilon$ . To obtain our lower bounds, we use the Gap Exponential Time Hypothesis (Gap-ETH) as well as a new hypothesis regarding the hardness of approximating the well known Densest  $\kappa$ -Subgraph problem in subexponential time (these hypotheses are used separately in proving different lower bounds). For example, we prove that under reasonable hardness assumptions, any *proper* learning algorithm for finding the best fitting ReLU must run in time exponential in  $1/\epsilon^2$ . Together with a previous work regarding improperly learning a ReLU [21], this implies the first separation between proper and improper algorithms for learning a ReLU. We also study the problem of properly learning a depth-2 network of ReLUs with bounded weights giving new (worst-case) upper bounds on the running time needed to learn such networks both in the realizable and agnostic settings. Our upper bounds on the running time essentially matches our lower bounds in terms of the dependency on  $\epsilon$ .

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Machine learning theory; Theory of computation  $\rightarrow$  Problems, reductions and completeness

**Keywords and phrases** ReLU, Learning Algorithm, Running Time Lower Bound

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.22

**Related Version** A full version of the paper is available at <http://arxiv.org/abs/2011.13550>.

**Funding** *Surbhi Goel*: Work was done while the author was a PhD student at UT Austin and was supported by the JP Morgan AI Research PhD Fellowship.

*Adam Klivans*: Supported by NSF awards AF-1909204, AF-1717896, and the NSF AI Institute for Foundations of Machine Learning (IFML). Work done while visiting the Institute for Advanced Study, Princeton, NJ.

---

\* This work subsumes our earlier manuscript [30].



© Surbhi Goel, Adam Klivans, Pasin Manurangsi, and Daniel Reichman;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 22; pp. 22:1–22:14



Leibniz International Proceedings in Informatics

LIPIcs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

*Pasin Manurangsi:* Part of this work was done while the author was at UC Berkeley and was partially supported by NSF under Grants No. CCF 1655215 and CCF 1815434.

**Acknowledgements** We would like to thank Amit Daniely, Amir Globerson, Meena Jagadeesan and Cameron Musco for interesting discussions.

## 1 Introduction

Neural networks have become popular in machine learning tasks arising in multiple applications such as computer vision, natural language processing, game playing and robotics [25]. One attractive feature of neural networks is being universal approximations: a network with a single hidden layer<sup>1</sup> with sufficiently many neurons can approximate arbitrary well any measurable real-valued function [23, 13]. These networks are typically trained on labeled data by setting the weights of the units to minimize the loss function (often the squared loss is used) over the training data. The challenge is to find a computationally efficient way to set the weights to achieve low error. While heuristics such as stochastic gradient descent (SGD) have been successful in practice, our theoretical understanding about the amount of running-time needed to train neural networks is still lacking.

It has been known for decades [8, 31, 24] that finding a set of weights that minimizes the loss of the training set is NP-hard. These hardness results, however, only apply to classification problems and to settings where the neural networks involved use discrete, Boolean activations. Our focus here is on neural networks with real inputs whose neurons have the real-valued ReLU activation function. Specifically, we consider depth-2 networks of ReLUs, namely either a single ReLU or a weighted sum of ReLUs<sup>2</sup>, and the optimization problem of training them given labeled data points, which are defined below.

► **Definition 1.** A rectifier is the real function  $[x]_+ := \max(0, x)$ . A rectified linear unit (ReLU) is a function  $f(\mathbf{x}) : \mathbb{R}^n \rightarrow \mathbb{R}$  of the form  $f(\mathbf{x}) = [\langle \mathbf{w}, \mathbf{x} \rangle]_+$  where  $\mathbf{w} \in \mathbb{R}^n$  is fixed. A depth-2 neural network with  $k$  ReLUs (abbreviated as  $k$ -ReLU) is a function from  $\mathbb{R}^n$  to  $\mathbb{R}$  defined by

$$\text{ReLU}_{\mathbf{w}^1, \dots, \mathbf{w}^k, \mathbf{a}}(\mathbf{z}) = \sum_{j=1}^k a_j [\langle \mathbf{w}^j, \mathbf{z} \rangle]_+.$$

Here  $\mathbf{x} \in \mathbb{R}^n$  is the input,  $\mathbf{a} = (a_1, \dots, a_k) \in \{-1, 1\}^k$  is a vector of “coefficients”,  $\mathbf{w}^j = (w_1^j, \dots, w_n^j) \in \mathbb{R}^n$  is a weight vector associated with the  $j$ -th unit. When  $a_1 = \dots = a_k = 1$ , we refer to  $\text{ReLU}_{\mathbf{w}^1, \dots, \mathbf{w}^k, \mathbf{a}}(\mathbf{z})$  as the sum of  $k$  ReLUs, and we may omit  $\mathbf{a}$  from the subscript.

We note that the assumption that  $a_1, \dots, a_k \in \{+1, -1\}$  is without loss of generality (e.g., [32]): for any non-zero  $a_1, \dots, a_k \in \mathbb{R} \setminus \{0\}$  and  $\mathbf{w}^1, \dots, \mathbf{w}^k$ , we may consider  $\hat{a}_1 = \frac{a_1}{|a_1|}, \dots, \hat{a}_k = \frac{a_k}{|a_k|}$  and  $\hat{\mathbf{w}}^1 = |a_1| \mathbf{w}^1, \dots, \hat{\mathbf{w}}^k = |a_k| \mathbf{w}^k$  instead, which represent the same depth-2 network of  $k$  ReLUs.

When training neural networks composed of ReLUs, a popular method is to find, given training data, a set of coefficients and weights for each gate minimizing the squared loss.

<sup>1</sup> We also refer to such networks as depth-2 networks or shallow networks.

<sup>2</sup> We also assume all the biases of the units are 0.

► **Definition 2.** Given a set of  $m$  samples  $\mathbf{x}_1, \dots, \mathbf{x}_m \in \mathbb{R}^n$  along with  $m$  labels  $y_1, \dots, y_m \in \mathbb{R}$ , our goal is to find  $\mathbf{w}^1, \dots, \mathbf{w}^k, \mathbf{a}$  which minimize the average squared training error of the sample, i.e.,

$$\min_{\mathbf{w}^1, \dots, \mathbf{w}^k, \mathbf{a}} \frac{1}{m} \sum_{i=1}^m (\text{ReLU}_{\mathbf{w}^1, \dots, \mathbf{w}^k, \mathbf{a}}(\mathbf{x}_i) - y_i)^2 \quad (1)$$

We refer to the optimization problem (1) as the  $k$ -ReLU training problem (aka  $k$ -ReLU regression).

When  $\mathbf{w}^j = (w_1^j, \dots, w_n^j)$  are assumed to have Euclidean norm at most 1 and  $y_i$  are assumed to be in  $[-k, k]$ , we refer to the optimization problem above as the bounded  $k$ -ReLU training problem.

Sometimes we assume that the “coefficient” vector  $\mathbf{a}$  is fixed in advance (and known to the optimizer) and not part of the input to the training problem. We mention this explicitly when relevant. Also observe that in the optimization problem above we are looking for a **global minimum** rather than a local minimum. A multiset of samples  $\{(\mathbf{x}_i, y_i)\}_{i \in [m]}$  is said to be *realizable* if there exist  $\mathbf{w}^1, \dots, \mathbf{w}^k, \mathbf{a}$  which result in zero training error.

Our goal is to pin down the computational complexity of the training problem for depth-2 networks of ReLUs, by answering the following question:

► **Question 3.** What is the worst-case running time of training a  $k$ -ReLU?

We focus on depth-2 networks which are rather involved and give rise to nontrivial algorithmic challenges [41, 6]. Understanding shallow networks seems to be a prerequisite for understanding the complexity of training networks of depth greater than 2.

## 1.1 Our results

We first consider arguably the simplest possible network: a single ReLU. We show that, already for such a network, the training problem is NP-hard. In fact, our result even rules out a large factor *multiplicative* approximation of the minimum squared error, as stated below.

► **Theorem 4** (Hardness of Training a single ReLU). *The 1-ReLU training problem is NP-hard. Furthermore, given a sample of  $m$  data points of dimension  $n$  it is NP-hard to approximate the optimal squared error within a multiplicative factor of  $(nm)^{1/\text{poly} \log \log(nm)}$ .*

Given such a strong multiplicative inapproximability result, a natural question is whether one can get a good algorithm for *additive* approximation guarantee. Notice that we cannot hope for additive approximation in general, because scaling the samples and their labels can make the additive approximation gap arbitrarily large. Hence, we must consider the bounded 1-ReLU Training problem. For this, we give a simple  $2^{O(1/\epsilon^2)} \text{poly}(n, m)$  time algorithm with additive approximation  $\epsilon$ . Furthermore, it easily generalizes to the case of the bounded  $k$ -ReLU Training problem for  $k > 1$ , but we have to pay a factor of  $k^5$  in the exponent:

► **Theorem 5** (Training Algorithm). *There is a (randomized) algorithm that can solve the bounded  $k$ -ReLU training problem to within any additive error  $\epsilon > 0$  in time  $2^{O(k^5/\epsilon^2)} \text{poly}(n, m)$ .*

Perhaps more surprisingly, we can prove a tight running time lower bound for the bounded 1-ReLU training problem, which shows that the term  $1/\epsilon^2$  in the exponent is necessary. Our running time lower bound relies on the assumption that there is no subexponential

time algorithm for approximating the *Densest  $\kappa$ -Subgraph* problem within any constant (multiplicative) factor. Recall that, in the Densest  $\kappa$ -Subgraph (D $\kappa$ S) problem, we are given a graph  $G = (V, E)$  and a positive integer  $\kappa$ . The goal is to select a subset  $T \subseteq V$  of  $\kappa$  vertices that induces as many edges as possible. We use  $\text{den}_\kappa(G)$  to denote this optimum<sup>3</sup> and  $N$  to denote the number of vertices,  $|V|$ . Our hypothesis can be stated formally as follows.

► **Hypothesis 6.** *For every constant  $C \geq 1$ , there exist<sup>4</sup>  $\delta = \delta(C) > 0$  and  $d = d(C) \in \mathbb{N}$  such that the following holds. No  $O(2^{\delta N})$ -time algorithm can, given an instance  $(G, \kappa)$  of D $\kappa$ S where each vertex of  $G$  has degree at most  $d$  and an integer  $\ell$ , distinguish between the following two cases:*

- (Completeness)  $\text{den}_\kappa(G) \geq \ell$ .
- (Soundness)  $\text{den}_\kappa(G) < \ell/C$ .

While this hypothesis is new (we are the first to introduce it), it seems fair to say that refuting it will require a breakthrough in current algorithms for the D $\kappa$ S problem. There are also other supporting evidences for the validity of this hypothesis. For example, it is known that  $o(N)$ -level of the Sum-of-Squares Hierarchies do not give constant factor approximation for D $\kappa$ S even for bounded degree graphs [7, 12, 28]. Furthermore, these Sum-of-Squares lower bounds are proved via reductions from a certain family of random CSPs, whose Sum-of-Squares lower bounds are shown in [35, 40]. This means that, if Hypothesis 6 is false, then one can refute this family of sparse random CSPs in subexponential time. This would constitute an arguably surprising development in the area of refuting random CSPs, which has been extensively studied for decades (see [1] and references therein).

As mentioned earlier, assuming Hypothesis 6, we can prove the tight running time lower bound for the bounded 1-ReLU Training problem:

► **Theorem 7** (Tight Running Time Lower Bound for 1-ReLU Training). *Assuming Hypothesis 6, there is no algorithm that, for all given  $\epsilon > 0$ , can solve the bounded 1-ReLU training problem within an additive error  $\epsilon$  in time  $2^{o(1/\epsilon^2)} \text{poly}(n, m)$ .*

We remark that, akin to standard conventions in the area of fine-grained and parameterized complexity, all lower bounds are stated against algorithms that work for *all* values of  $\epsilon$  with the specified running time. Indeed, it is possible to significantly speed up the time bound  $2^{O(1/\epsilon^2)} \text{poly}(n, m)$  for extreme values of  $\epsilon$ ; for instance, enumerating all possible  $\mathbf{w}$  over a  $\Theta(\epsilon)$ -net<sup>5</sup> of  $\mathcal{B}^n$  gives an algorithm that runs in time  $O(1/\epsilon)^{O(n)} \text{poly}(m)$ , which is asymptotically smaller than  $2^{O(1/\epsilon^2)} \text{poly}(n, m)$  when  $\epsilon = o\left(\frac{1}{\sqrt{n \log n}}\right)$ . Nonetheless, our lower bounds can be extended to include a large range of “reasonable”  $\epsilon$ . Further discussion on such an extension is provided before Section 1.3.

An interesting consequence of Theorem 7 is that it gives a separation between proper and improper agnostic learning of 1-ReLU. Specifically, [21] shows that improper agnostic learning of 1-ReLU can be done in  $2^{O(1/\epsilon)} \text{poly}(n)$  time, while Theorem 7 rules out such a possibility for proper agnostic learning.

<sup>3</sup> Equivalently,  $\text{den}_\kappa(G) := \max_{T \subseteq V, |T|=\kappa} |E(T)|$ .

<sup>4</sup> As  $C$  increases,  $\delta$  and  $d$  decreases.

<sup>5</sup> Recall that an  $\delta$ -net (also refer to as an  $\delta$ -cover) of a set  $S \subseteq \mathbb{R}^n$  is a set  $T \subseteq \mathbb{R}^n$  such that, for every  $x \in S$ , there exists  $y \in T$  where  $\|x - y\|_2 \leq \delta$ . It is well-known that, for any  $\delta \in [0, 1]$ , there is a  $\delta$ -net of the unit ball  $\mathcal{B}^n$  of size  $(3/\delta)^n$  and that it can be found in  $(3/\delta)^{O(n)}$  time.

### Training $k$ -ReLU: The Realizable Case

An important special case of the  $k$ -ReLU Training problem is the realizable case, where there is an unknown  $k$ -ReLU that labels every training sample correctly. When  $k = 1$ , it is straightforward to see that the realizable case of 1-ReLU Training can be phrased as a linear program and hence can be solved in polynomial time. On the other hand, we show that, once  $k > 1$ , the problem becomes NP-hard:

► **Theorem 8** (Hardness of Training  $k$ -ReLU in the Realizable Case). *For any constant  $k \geq 2$ , the  $k$ -ReLU training problem is NP-hard even in the realizable case.*

Our result is in fact slightly stronger than stated above: specifically, we show that, when the samples can be realizable by a (non-negative) sum of  $k$  ReLUs (i.e.  $k$ -ReLU when  $\mathbf{a}$  is the all-one vector), it is still NP-hard to find a  $k$ -ReLU that realizes the samples even if negative coefficients in  $\mathbf{a}$  are allowed. Furthermore, while we assume in this theorem that  $k$  is a constant independent of  $n$ , one can also prove an analogous hardness result, when  $k$  grows sufficiently slowly as a function of  $n$ . We refer the full version for more details.

Observe that Theorem 8 implies that efficient *multiplicative* approximation for the  $k$ -ReLU Training problem is impossible (assuming  $P \neq NP$ ) for  $k \geq 2$ . As a result, we once again turn to additive approximation. On this front, we can improve the running time of the algorithm in Theorem 5 when we assume that the samples are realizable, as stated below.

► **Theorem 9** (Training Algorithm in the Realizable Case). *When the given samples are realizable by some  $k$ -ReLU, there is a (randomized) algorithm that can solve the bounded  $k$ -ReLU training problem to within any additive error  $\epsilon > 0$  in time  $2^{O((k^3/\epsilon) \log^3(k/\epsilon))} \text{poly}(n, m)$ .*

Importantly, the dependency of  $\epsilon$  in the exponent is  $\tilde{O}(1/\epsilon)$ , instead of  $1/\epsilon^2$  that appeared in the non-realizable case (i.e. Theorems 5 and 7). We can also show that this dependency is tight (up to log factors), in the realizable case, under the Gap Exponential Time Hypothesis (Gap-ETH) [17, 29], a standard complexity theoretic assumption in parameterized complexity (see e.g. [11]). Gap-ETH states that there exists  $\delta > 0$  such that no  $2^{o(n)}$ -time algorithm can, given a CNF formula with  $n$  Boolean variables, distinguish between (i) the case where the formula is satisfiable, and (ii) the case where any assignment violates at least  $\delta$  fraction of the clauses. Our running time lower bound can be stated more formally as follows.

► **Theorem 10** (Tight Running Time Lower Bound for the Realizable Case). *Assuming Gap-ETH, for any constant  $k \geq 2$ , there is no algorithm that, for all given  $\epsilon > 0$ , can solve the bounded  $k$ -ReLU training problem within an additive error  $\epsilon$  in time  $2^{o(1/\epsilon)} \text{poly}(n, m)$  even when the input samples are realizable by some  $k$ -ReLU.*

### Relation to Learning ReLUs

$k$ -ReLU Training is closely related to the problem of *proper learning* of  $k$ -ReLU. In fact, an algorithm for the latter also solves the former. Hence, our hardness results immediately implies hardness of proper learning of  $k$ -ReLU as well. Furthermore, our algorithm also works for the learning problem. Please refer to the full version for more details.

### Stronger Quantifier in Running Time Lower Bounds

As stated earlier, our running time lower bounds in Theorems 7 and 10 hold only against algorithms that work *for all*  $\epsilon > 0$ . A natural question is whether one can prove lower bounds against algorithms that work only *for some* “reasonable” values of  $\epsilon$ . As explained in more detail below, we can quite easily also get a lower bound with this latter (stronger) quantifier, for any “reasonable” value of  $\epsilon$ .

First, our lower bounds in Theorems 7 and 10 both apply in the regime where the lower bounds themselves are  $2^{\Theta(n)}$ ; in other words,  $\epsilon = \Theta(1/\sqrt{n})$  in Theorem 7 and  $\epsilon = \Theta(1/n)$  in Theorem 10. These are essentially the smallest possible value of  $\epsilon$  for which the lower bounds in Theorems 7 and 10 can hold, because the aforementioned algorithm that enumerates over an  $\epsilon$ -net of  $\mathcal{B}^n$  solves the problem in time  $O(1/\epsilon)^{O(n)} \text{poly}(n)$ . On the other hand, for smaller values of  $\epsilon$ , we can get a running time lower bound easily by “padding” the dimension by “dummy” coordinates that are always zero. For instance, if we start with  $\epsilon = \Theta(1/\sqrt{n})$ , then we may pad the instance to say  $n' = n^2$  dimensions, resulting in the relationship  $\epsilon = \Theta(1/\sqrt{n'})$ . To summarize, this simple padding technique immediately gives the following stronger quantifier version of Theorem 7:

► **Theorem 11.** *For any non-increasing and efficiently computable<sup>6</sup> function  $\epsilon : \mathbb{N} \rightarrow \mathbb{R}^+$  such that  $\omega(\sqrt{\log n}) \leq \frac{1}{\epsilon(n)} \leq o(\sqrt{n})$ , assuming Hypothesis 6, there is no algorithm that can solve the bounded 1-ReLU training problem within an additive error  $\epsilon(n)$  in time  $2^{o(1/\epsilon(n)^2)} \text{poly}(n, m)$ .*

Notice that the constraint  $\omega(\sqrt{\log n}) \leq \frac{1}{\epsilon(n)}$  is also essentially necessary, because for  $\epsilon > \sqrt{\frac{\log \log n}{\log n}}$  our algorithm (Theorem 5) already runs in polynomial time. A strong quantifier version of Theorem 10 similar to above can be shown as well (but with  $\omega(\log n) \leq \frac{1}{\epsilon(n)} \leq o(n)$ ). We omit the full (straightforward) proof via padding of Theorem 11; interested readers may refer to the proof of Lemma 3.4 of [16] which employs the same padding technique.

## 1.2 Independent and concurrent work

There have been several concurrent and independent works to ours that we mention here. We remark that the techniques in these works are markedly different than the ones in this paper. For a single ReLU, [14] proved that the 1-ReLU Training problem is NP-hard. With respect to two ReLUs, [6] showed that finding weights minimizing the squared error of a 2-ReLU is NP-hard, even in the realizable case. The work of [9] considered the problem of training a network with a slightly different architecture, in which there are two ReLUs in the first hidden layer and the final output gate is also a ReLU (instead of a sum gate as in our case); they showed that, for such networks with three ReLUs (two in the hidden layer, one in the output layer), the training problem is NP-hard even for the realizable case. As a result of having an output gate computing a ReLU, our NP-hardness result (regarding training a sum of two ReLUs) does not imply their result and their hardness result does not imply our hardness result for training a sum of 2 ReLUs.

## 1.3 Related work

The computational aspects of training and learning neural networks has been extensively studied. Due to this, we only focus on those directly related to our results.

We are not aware of a previous work showing that the general  $k$ -ReLU training problem is NP-hard for  $k > 2$ , nor are we aware of previous results regarding the hardness of approximating the squared error of a single ReLU. The  $k = 2$  case and the  $k > 2$  case seem to require different ideas and indeed our proof technique for Theorem 10 is different than those of [9, 6]. Moreover, the question of generalizing the NP-hardness result from  $k = 2$  to  $k > 2$  is mentioned explicitly in [9]. Finally, we remark that neither [14] nor [6] provides explicit

---

<sup>6</sup> That is, we assume that computing  $\epsilon(n)$  can be done in time  $\text{poly}(n)$  for any  $n \in \mathbb{N}$ .



running time lower bounds in terms of  $1/\epsilon$  for the problem of training  $k$  ReLUs within an additive error of  $\epsilon$ . To the best of our knowledge, our work is the first to obtain such lower bounds.

[42] has proven that finding weights minimizing the squared error of a  $k$ -ReLU is NP-hard when  $\mathbf{a}$  is the all-one vector (or alternatively, when all the coefficients of the units are restricted to be positive) for every  $k \geq 2$ .

Some sources (e.g. [4, 5]) attribute (either implicitly or explicitly) the NP-hardness of the  $k$ -ReLU Training problem to [8], who consider training a neural network with *threshold units*. However, it is unclear (to us) how to derive the NP-hardness of training ReLUs from the hardness results of [8]. Several NP-hardness results for training neural networks with architectures differing from the fully connected architecture considered here are known. For example, in [10], the training problem is shown to be hard for a depth-2 *convolutional network* with (at least two) *non-overlapping* patches. To the best of our knowledge, these architectural differences render those previous results inapplicable for deriving the hardness results regarding the networks considered in this work.

Several papers have studied a slightly different setting of *improper learning* of neural networks. An example is [26] who show that improper learning of depth-2 networks of  $\omega(1)$  ReLUs is hard, assuming certain average case assumptions. More recently, [21] show that even for a single ReLU, when  $|\langle \mathbf{w}, \mathbf{x} \rangle|$  tends to infinity with  $n$ , learning  $[\langle \mathbf{w}, \mathbf{x} \rangle]_+$  improperly in time  $g(\epsilon) \cdot \text{poly}(n)$  is unlikely as it will result in an efficient algorithm for the problem of learning sparse parities with noise which is believed to be intractable. These hardness results for improper learning do imply hardness for the corresponding training problems. Nonetheless, it should be noted that the fact that these results have to rely on assumptions other than  $P \neq NP$  is not a coincidence: it is known that basing hardness of improper learning on  $P \neq NP$  alone will result in a collapse of the Polynomial Hierarchy [3].

On the algorithmic side, Arora et al. [4] provide a simple and elegant algorithm that exactly solves the ReLU training problem in polynomial time assuming the dimension  $n$  of the data points is an absolute constant; Arora et al.'s algorithm is for the networks we consider, and it has since been also extended to other types of networks [9]. Additionally, there have also been works on (agnostic) learning algorithms for ReLUs. Specifically, Goel et al. [21] consider the bounded norm setting where the inputs to the ReLUs as well as the weight vectors of the units have norms at most 1. For this setting, building on kernel methods and tools from approximation theory, they show how to *improperly* learn a single  $n$ -variable ReLU up to an additive error of  $\epsilon$  in time  $2^{O(1/\epsilon)} \cdot \text{poly}(n)$ . Their result generalizes to depth-2 ReLUs with  $k$  units with running time of  $2^{O(\sqrt{k}/\epsilon)} \cdot \text{poly}(n)$  assuming the coefficient vector  $\mathbf{a}$  has norm at most 1. The algorithm they provide is quite general: it works for arbitrary distribution over input-output pairs, for  $\epsilon$  that can be small as  $1/\log n$  and also for the reliable setting.

A limitation of our hardness results is that they consider “pathological” training data sets that are specifically constructed to encode intractable combinatorial optimization problems. Several works in literature have tried to overcome this issue by considering the training/learning problems on more “benign” data distributions, such as log-concave distributions or those with Gaussian marginals. On this front, both algorithms and lower bounds have been shown for depth-2 networks [38, 6, 22].

Using insights from the study of exponential time algorithms towards understanding the complexity of machine learning problems as is done in this work is receiving attention lately [36, 15, 37].



## 1.4 Organization of the Paper

In the remainder of the main body of this paper, we provide high-level overviews of our proofs (Section 2) and discuss several potential research directions (Section 3). Due to space constraints, all proofs are deferred to the full version.

## 2 Proof Overview

Below we provide the informal overviews of our proofs and intuition behind them.

### NP-Hardness of Training 1-ReLU

Our reduction is from the (NP-hard) Set Cover problem, in which we are given subsets  $T_1, \dots, T_M$  of a universe  $U$ , and the goal is to select as few of these subsets as possible whose union covers the entire universe  $U$ . We reduce this to the problem of 1-ReLU Training, where the dimension  $n$  is equal to  $M$ . We think of each coordinate of  $\mathbf{w}$  as an unknown (i.e. variable); specifically, the desired solution will have  $w_i = -1$  iff  $T_i$  is picked and 0 otherwise. From this perspective, adding a labelled sample  $(\mathbf{x}, y)$  is the same as adding a “constraint”  $[\mathbf{w} \cdot \mathbf{x}]_+ = y$ . There are two types of constraints we will add:

- (Element Constraint) For each  $u \in U$ , we add a constraint of the form  $[1 + \sum_{T_i \ni u} w_i]_+ = 0$ . The point is that such a constraint is satisfied when  $u$  is covered by the selected subsets.
- (Subset Constraint) For each  $i \in [M]$ , we add a constraint of the form  $[\gamma + w_i]_+ = \gamma$  for some small  $\gamma > 0$ . This constraint will be violated for any selected subset.

By balancing the weights (i.e. number of copies) of each constraint carefully, we can ensure that the element constraints are never unsatisfied, and that the goal is ultimately to violate as few subset constraints as possible, which is equivalent to trying to pick as few subsets as possible that can fully cover  $U$ . This completes the high-level overview of our reduction.

We remark that there is a subtle point here because we cannot directly have a constant such that 1 or  $\gamma$  in the constraints themselves. Rather, we need to have “constraint coordinate” and adding the constants through this coordinate. This will also be done in the other reductions presented below, and we will not mention this again.

The outlined proof, together with the  $\Theta(\log |U|)$  inapproximability of Set Cover [27, 20], already gives a hardness of approximation of a multiplicative factor of  $\Theta(\log(nm))$  for the 1-ReLU Training problem. To further improve this inapproximability ratio to  $(nm)^{1/\text{poly} \log \log(nm)}$ , we reduce from the *Minimum Monotone Circuit Satisfiability* (MMCS) problem, which is a generalization of Set Cover. In MMCS, we are given a monotone circuit and the goal is to set as few input wires to true as possible under the condition that the circuit’s output must be true. Strong inapproximability results for MMCS are known (e.g. [18]). Our reduction from MMCS proceeds in a similar manner as that of the Set Cover reduction above. Roughly speaking, the modification is that each unknown is now whether each wire is set/evaluated to true, whereas the constraints are now to ensure that the evaluation at each gate is correct and that the output is true.

### Tight Running Time Hardness of 1-ReLU Training

We now move on to the proof overview of the tight running time lower bound for 1-ReLU Training. Recall that we will be reducing from the Densest  $\kappa$ -Subgraph ( $\text{D}\kappa\text{S}$ ) problem, in which we are given a graph  $G = (V, E)$  and  $\kappa \in \mathbb{N}$ . The goal is to find a set of  $\kappa$  vertices that induces the maximum number of edges.

To motivate our construction, a simple combination of dimensionality reduction and  $\delta$ -net can in fact find a ReLU that point-wise approximates the optimal ReLU to within an additive factor of  $\delta$  in time  $2^{\tilde{O}(1/\delta^2)} \text{poly}(n)$ . That is, if the ReLU that achieves the optimal error has weight vector  $\mathbf{w}^*$ , then we can find a weight vector  $\mathbf{w}$  such that  $|\mathbf{w} \cdot \mathbf{x}|_+ - [\mathbf{w}^* \cdot \mathbf{x}]_+| \leq \delta$  for all input samples  $(\mathbf{x}, y)$  in time<sup>7</sup>  $2^{\tilde{O}(1/\delta^2)} \text{poly}(n)$ .

Indeed, this is an explanation why, in the realizable case, we can get  $\epsilon$  squared error in  $2^{\tilde{O}(1/\delta)} \text{poly}(n)$  time by simply picking  $\delta = \sqrt{\epsilon}$ . Now, since we need our hardness here (for the non-realizable case) to hold with stronger running time lower bound of  $2^{\Theta(1/\epsilon^2)} \text{poly}(n)$ , we have to make sure that whenever  $\delta \gg \epsilon$ , the aforementioned point-wise approximation of  $\delta$  is not sufficient to get an error of  $\epsilon$ . Suppose that, for an input labelled sample  $(\mathbf{x}, y)$ , the optimal ReLU outputs  $y'$  and our approximation outputs  $y''$  (where  $|y'' - y'| \leq \delta$ ). Notice that the difference in the square error between the two for this sample is only at most  $O((y' - y)\delta) + \delta^2$ . Now, if we want this quantity to be at least  $\epsilon$  for any  $\delta \geq \Omega(\epsilon)$ , then it must be that  $|y' - y| = \Omega(1)$ . In other words, we have to make our samples so that even the optimal ReLU is “wrong” by  $\Omega(1)$  additive factor (on average); this indeed means that, if the ReLU we find is “more wrong” by an additive factor of  $\Theta(\epsilon)$ , then the increase in the average squared error would be  $\Omega(\epsilon)$  as desired.

With the observation in the previous paragraph in mind, we will now provide a rough description of our gadget. Given a  $\text{D}\kappa\text{S}$  instance  $(G = (V, E), \kappa)$ , our samples will have  $|V|$  dimensions, one corresponding to each vertex. In the YES case where there is  $T \subseteq V$  of size  $\kappa$  that induces many edges, we aim to have our ReLU weight assigning  $\frac{1}{\sqrt{\kappa}}$  to all coordinates corresponding to vertices in  $T$ , and zero to all other coordinates. To enforce this, we first add a sample for every vertex  $v \in V$  that corresponds to the constraint

$$\left[ \mathbf{w}_v - \frac{1}{2\sqrt{\kappa}} \right]_+ = 1.$$

We refer to these as the *cardinality constraints*. While this may look peculiar at first glance, the effect is that it ensures that roughly speaking  $\mathbf{w}$  has  $\kappa$  coordinates that are “approximately”  $\frac{1}{\sqrt{\kappa}}$  and the remaining coordinates are “small”. To see that this is the case, observe that the average mean squared error here is  $1 - \frac{2}{|V|} \sum_{v \in V} \left[ \mathbf{w}_v - \frac{1}{2\sqrt{\kappa}} \right]_+ + \frac{1}{|V|} \sum_{v \in V} \left[ \mathbf{w}_v - \frac{1}{2\sqrt{\kappa}} \right]_+^2$ . The last term is small and may be neglected. Hence, we essentially have to maximize  $\sum_{v \in V} \left[ \mathbf{w}_v - \frac{1}{2\sqrt{\kappa}} \right]_+$ . This term is indeed maximized when  $\mathbf{w}$  has  $\kappa$  coordinates equal to  $\frac{1}{\sqrt{\kappa}}$ , and zeros in the remaining coordinates. Notice here that this also fits with our intuition from the previous paragraph: even in the optimal ReLU, the value out put by the value (which is either 0 or  $\frac{1}{2\sqrt{\kappa}}$ ) is  $\Omega(1)$  away from the input label of the sample (i.e. 1).

So far, the cardinality constraints have ensured that  $\mathbf{w}$  “represents” a set  $T \subseteq V$  of size roughly  $\kappa$ . However, we have not used the fact that  $T$  contains many edges at all. Thus, for every edge  $e = \{u, v\} \in E$ , we also add the example corresponding to the following constraint to our distribution:

$$\frac{1}{2} \left[ \mathbf{w}_u + \mathbf{w}_v - \frac{1.75}{\sqrt{\kappa}} \right]_+ = 1.$$

We call these the *edge constraints*. The point here is that, if  $e$  is not an induced edge in  $T$ , then the output of the ReLU will be zero. On the other hand, if  $e$  is an edge in  $T$ , then the output of the ReLU will be  $\frac{0.25}{\sqrt{\kappa}}$ . Hence, the more edges  $T$  induces, the smaller the error.

<sup>7</sup> We assume throughout that  $m = \text{poly}(1/\delta)$ , which is w.l.o.g. due to standard generalization bounds.

By carefully selecting weights (i.e. number of copies) of each sample, one can indeed show that the average square error incurred in the completeness and soundness case of Hypothesis 6 differs by  $\epsilon = \Omega\left(\frac{1}{\sqrt{|V|}}\right)$ . Hence, if we can solve the 1-ReLU Training problem to within an additive error of  $\epsilon$  in time  $2^{o(1/\epsilon^2)}\text{poly}(n, m)$ , we can also solve the problem in Hypothesis 6 in time  $2^{o(|V|)}$ , which breaks the hypothesis.

### Hardness of Training $k$ -ReLU in the Realizable Case

We next consider the problems of Training  $k$ -ReLU for  $k \geq 2$  in the realizable case. Both the NP-hardness result (Theorem 8) and the tight running time lower bound (Theorem 10) employ similar reductions. These reductions proceed in two steps. First is to reduce from the NP-hard  $k$ -coloring problem to the problem of training *non-negative sum of  $k$  ReLUs*, in which we fix the coefficient vector  $\mathbf{a}$  to be the all-one vector and only seeks to find  $\mathbf{w}^1, \dots, \mathbf{w}^k$  that minimizes the squared error. Then, in the second step, we reduce this to the original problem of  $k$ -ReLU Training (where the coefficient vector  $\mathbf{a}$  can be negative).

*Step I: From  $k$ -Coloring to Training Sum of  $k$  ReLUs.* The NP-hardness of Sum of  $k$  ReLUs Training in fact follows directly from a reduction of [42]. We will now sketch Vu's reduction, since it will be helpful in our subsequent discussions below. Vu's reduction starts from the  $k$ -coloring problem, in which we are given a hypergraph  $G = (V, E)$  and the goal is to determine whether there is a proper  $k$ -coloring<sup>8</sup> of the hypergraph. Given an instance  $G = (V, E)$  of  $k$ -coloring, the number of dimensions in the training problem will be  $n = |V|$  where we associate each dimension with a vertex. Notice that now we have  $k$  unknowns associated to each vertex  $v$ :  $w_v^1, \dots, w_v^k$ . In the desired solution, these variables will tell us which color  $v$  is assigned to: specifically,  $w_v^i > 0$  iff  $v$  is colored  $i$  and  $w_v^i \leq 0$  otherwise.

Adding a labelled sample  $(\mathbf{x}, y)$  is the same as adding a "constraint"  $[\mathbf{w}^1 \cdot \mathbf{x}]_+ + \dots + [\mathbf{w}^k \cdot \mathbf{x}]_+ = y$ . There are two types of constraints we will add:

- (Vertex Constraint) For every vertex  $v \in V$ , we add a constraint<sup>9</sup>  $[w_v^1]_+ + \dots + [w_v^k]_+ = 1$ . This constraint ensures that, for every  $v \in V$ , we must have  $w_v^{i_v} > 0$  for at least one  $i_v \in [k]$ , meaning that the vertex  $v$  is assigned at least one color.
- (Hyperedge Constraint) For every hyperedge  $e = \{v_1, \dots, v_\ell\} \in E$ , we add a constraint<sup>10</sup>  $[w_{v_1}^1 + \dots + w_{v_\ell}^1]_+ + \dots + [w_{v_1}^k + \dots + w_{v_\ell}^k]_+ = 0$ . This ensures that the hyperedge  $e$  is not monochromatic. Otherwise, we have  $i_{v_1} = \dots = i_{v_\ell}$  meaning that  $w_{v_1}^{i_{v_1}} + \dots + w_{v_\ell}^{i_{v_1}} > 0$ , which violates the hyperedge constraint.

This finishes our summary of Vu's reduction, which gives the NP-hardness of training a (non-negative) sum of  $k$  ReLUs.

*Step II: Handling Negative Coefficients.* The argument above, especially for the hyperedge constraints, relies on the fact that the coefficient vector  $\mathbf{a}$  is the all-one vector. In other words, even if the input hypergraph is not  $k$ -coloring, it is still possible that there is a  $k$ -ReLU (possibly negative weight vector  $\mathbf{a}$ ) that realizes the samples. Hence, the reduction above does not yet work for our original problem of  $k$ -ReLU Training. To handle this issue, we use an additional gadget which is simply a set of labelled samples with the following properties: these samples can be realized by a  $k$ -ReLU only when the weight vectors  $\mathbf{a}$  is the

<sup>8</sup> A proper  $k$ -coloring is a mapping  $\chi : V \rightarrow [k]$  such that no hyperedge is monochromatic, or equivalently  $|\chi(e)| > 1$  for all  $e \in E$ .

<sup>9</sup> This constraint corresponds to  $\mathbf{x}$  being the  $v$ -th vector in the standard basis and  $y = 1$ .

<sup>10</sup> This constraint corresponds to  $\mathbf{x}$  being the indicator vector of  $e$  and  $y = 0$ .

all-one vector. Essentially speaking, by adding these samples also to our sample set, we have forced  $\mathbf{a}$  to be the all-one vector, at which point we restrict ourselves back to the case of (non-negative) sum of  $k$  ReLUs and we can use the hard instance from the above reduction from  $k$ -coloring. These are the main ideas of the proof of Theorem 8.

*Tight Running Time Lower Bound.* As stated earlier, the tight running time lower bound for the bounded  $k$ -ReLU Training problem (Theorem 10) follows from a similar reduction, except that we now have to (1) carefully select the number of copies of each sample and (2) scale the labels  $y_i$ 's down so that the norm of each of  $\mathbf{w}^1, \dots, \mathbf{w}^k$  is at most one. Roughly speaking, this means that the labels for the vertex constraints become  $\Theta(1/\sqrt{|V|})$  instead of 1 as before. In other words, each violated constraint roughly contributes to  $\Theta(1/|V|)$  squared error. Since it is known (assuming Gap-ETH) that distinguishing between a  $k$ -colorable hypergraph and a hypergraph for which every  $k$ -coloring violates a constant fraction of the edges takes  $2^{\Omega(|V|)}$  time (e.g. [33]), we can arrive at the conclusion that solving the bounded  $k$ -ReLU Training problem to within an additive squared error of  $\epsilon = \Theta(1/|V|)$  must take  $2^{\Omega(1/|V|)} = 2^{\Omega(1/\epsilon)}$  time as desired.

We remark here that, interestingly, [42] used the reduction from  $k$ -coloring only for the case of  $k = 2$  units and employed an additional gadget to handle the case  $k > 2$ . To the best of our knowledge, this approach seems to decrease the resulting error  $\epsilon$ , which means that the running time lower bound is not of the form  $2^{\Omega(1/\epsilon)}$ . On the other hand, we argue the hardness directly from  $k$ -coloring for any constant  $k \geq 2$ . This, together with a careful selection of the number of copies of each sample, allows us to achieve the running time lower bound in Theorem 10.

## Training and Learning Algorithms

Our  $k$ -ReLU training algorithm is based on the approach of [4]. The main idea behind the algorithm is to iterate over all possible sign patterns (whether each ReLU is active or not) of the inputs and subsequently solve the so formed convex optimization for each fixed pattern. The best hypothesis over all different sign patterns is chosen as the final hypothesis. It is not hard to see that the run-time for such an algorithm would be  $2^{(m+1)k} \text{poly}(n)$  since there are  $2^{mk}$  different sign patterns.

Using standard generalization bounds, one can show that the number of samples  $m$  needed for the empirical loss to be  $\epsilon$  close to the true loss is at most  $O(k^4/\epsilon^2)$ . Plugging this into the above algorithm gets us the desired running time ( $2^{O(k^5/\epsilon^2)} \text{poly}(n, m)$  as in Theorem 5) for the agnostic setting. For the realizable setting, we use an improved generalization result of [39], which implies that  $m = \tilde{O}(k^2/\epsilon)$  suffices; plugging this into the above algorithm yields us Theorem 9.

## 3 Conclusions and Open Questions

We have studied the computational complexity of training depth-2 networks with the ReLU activation function providing both NP-hardness results and algorithms for training ReLU's. Along the way we have introduced and used a new hypothesis regarding the hardness of approximating the Denset  $\kappa$ -Subgraph problem in subexponential time that may find applications in other settings. Our results provide a separation between proper and improper learning showing that for a single ReLU, proper learning is likely to be harder than improper learning. Our hardness results regarding properly learning shallow networks suggest that improperly learning such networks (for example, learning overparametrized networks whose number of units far exceeds the dimension of the labeled vectors [2, 19]) might be necessary to allow for tractable learning problems.

We stress here that our hardness results apply to minimizing the population loss<sup>11</sup> as well, since one may simply create an instance where the population is just the training data. Furthermore, the standard procedure for training neural networks is to perform ERM which is essentially minimizing the training loss. In fact, a bulk of theoretical work in the field focuses on generalization error assuming training error is small (often 0). Therefore, we believe it is a natural question to study the hardness of minimizing training loss.

Neural networks offer many choices (e.g., number of units, depth, choice of activation function, weight restrictions). Indicating which architectures are NP-hard to train can prove useful in guiding the search for a mathematical model of networks that can be trained efficiently. It should be remembered that our NP-hardness results are worst-case. Therefore they do not preclude efficient algorithms under additional distributional or structural assumptions [34]. Finally, as we focus on networks having significantly fewer units than data-points, the NP-hardness results reported here are not at odds with the ability to train neural networks in the overparameterized regime where there are polynomial time algorithms that can fit the data with zero error [43].

While we restrict our attention to algorithms for training networks with bounded weights, our exponential dependency of the running time on  $k$  (the number of units) makes these algorithms impractical. It remains an interesting question whether the dependency of the running time on  $k$  can be improved, or alternatively whether strong running time lower bounds can be shown in terms of  $k$  (similar to what is done for  $\epsilon$  in this work).

While we have focused on depth-2 networks, algorithms and lower bounds for deeper networks are of interest as well, especially given the multitude of their practical applications. It would be interesting to see whether the algorithms and hardness results extend to the setting of depth greater than 1. An interesting concrete question here is whether training/learning becomes harder as the network becomes deeper. For instance, is it possible to prove running time lower bounds that grow with the depth of the network?

---

## References

- 1 Sarah R. Allen, Ryan O'Donnell, and David Witmer. How to refute a random CSP. In *FOCS*, pages 689–708, 2015. doi:10.1109/FOCS.2015.48.
- 2 Zeyuan Allen-Zhu, Yuanzhi Li, and Yingyu Liang. Learning and generalization in overparameterized neural networks, going beyond two layers. In *Advances in neural information processing systems*, pages 6155–6166, 2019.
- 3 Benny Applebaum, Boaz Barak, and David Xiao. On basing lower-bounds for learning on worst-case assumptions. In *FOCS*, pages 211–220, 2008. doi:10.1109/FOCS.2008.35.
- 4 Raman Arora, Amitabh Basu, Poorya Mianjy, and Anirbit Mukherjee. Understanding deep neural networks with rectified linear units. In *International Conference on Learning Representations*, 2018. URL: [https://openreview.net/forum?id=B1J\\_rgWRW](https://openreview.net/forum?id=B1J_rgWRW).
- 5 Francis Bach. Breaking the curse of dimensionality with convex neural networks. *The Journal of Machine Learning Research*, 18(1):629–681, 2017.
- 6 Ainesh Bakshi, Rajesh Jayaram, and David P Woodruff. Learning two layer rectified neural networks in polynomial time. In *Conference on Learning Theory*, pages 195–268, 2019.
- 7 Aditya Bhaskara, Moses Charikar, Aravindan Vijayaraghavan, Venkatesan Guruswami, and Yuan Zhou. Polynomial integrality gaps for strong SDP relaxations of densest  $k$ -subgraph. In *SODA*, pages 388–405, 2012.
- 8 Avrim Blum and Ronald L Rivest. Training a 3-node neural network is NP-complete. In *Advances in neural information processing systems*, pages 494–501, 1989.

---

<sup>11</sup>The population loss is the expected square loss with respect to the distribution of data points.

- 9 Digvijay Boob, Santanu S Dey, and Guanghai Lan. Complexity of training relu neural network. *arXiv preprint*, 2018. [arXiv:1809.10787](#).
- 10 Alon Brutzkus and Amir Globerson. Globally optimal gradient descent for a convnet with gaussian inputs. *arXiv preprint*, 2017. [arXiv:1702.07966](#).
- 11 Parinya Chalermsook, Marek Cygan, Guy Kortsarz, Bundit Laekhanukit, Pasin Manurangsi, Danupon Nanongkai, and Luca Trevisan. From Gap-ETH to FPT-inapproximability: Clique, dominating set, and more. In *FOCS*, pages 743–754, 2017. doi:10.1109/FOCS.2017.74.
- 12 Eden Chlamtác, Pasin Manurangsi, Dana Moshkovitz, and Aravindan Vijayaraghavan. Approximation algorithms for label cover and the log-density threshold. In *SODA*, pages 900–919, 2017. doi:10.1137/1.9781611974782.57.
- 13 George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.
- 14 Santanu S Dey, Guanyi Wang, and Yao Xie. An approximation algorithm for training one-node relu neural network. *arXiv preprint*, 2018. [arXiv:1810.03592](#).
- 15 Ilias Diakonikolas, Daniel Kane, and Pasin Manurangsi. Nearly tight bounds for robust proper learning of halfspaces with a margin. In *Advances in Neural Information Processing Systems*, pages 10473–10484, 2019.
- 16 Ilias Diakonikolas, Daniel M. Kane, and Pasin Manurangsi. Nearly tight bounds for robust proper learning of halfspaces with a margin. *CoRR*, abs/1908.11335, 2019. [arXiv:1908.11335](#).
- 17 Irit Dinur. Mildly exponential reduction from gap 3SAT to polynomial-gap label-cover. *Electronic Colloquium on Computational Complexity (ECCC)*, 23:128, 2016. URL: <http://eccc.hpi-web.de/report/2016/128>.
- 18 Irit Dinur, Prahladh Harsha, and Guy Kindler. Polynomially low error PCPs with polyloglog  $n$  queries via modular composition. In *STOC*, pages 267–276, 2015. doi:10.1145/2746539.2746630.
- 19 Simon S Du, Jason D Lee, Haochuan Li, Liwei Wang, and Xiyu Zhai. Gradient descent finds global minima of deep neural networks. *arXiv preprint*, 2018. [arXiv:1811.03804](#).
- 20 Uriel Feige. A threshold of  $\ln n$  for approximating set cover. *J. ACM*, 45(4):634–652, 1998. doi:10.1145/285055.285059.
- 21 Surbhi Goel, Varun Kanade, Adam R. Klivans, and Justin Thaler. Reliably learning the relu in polynomial time. In *COLT*, pages 1004–1042, 2017. URL: <http://proceedings.mlr.press/v65/goel17a.html>.
- 22 Surbhi Goel, Sushrut Karmalkar, and Adam Klivans. Time/accuracy tradeoffs for learning a relu with respect to gaussian marginals. In *Advances in Neural Information Processing Systems*, pages 8582–8591, 2019.
- 23 Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.
- 24 Stephen Judd. On the complexity of loading shallow neural networks. *Journal of Complexity*, 4:177–192, 1988.
- 25 Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.
- 26 Roi Livni, Shai Shalev-Shwartz, and Ohad Shamir. On the computational efficiency of training neural networks. In *Advances in Neural Information Processing Systems*, pages 855–863, 2014.
- 27 Carsten Lund and Mihalis Yannakakis. On the hardness of approximating minimization problems. *J. ACM*, 41(5):960–981, 1994. doi:10.1145/185675.306789.
- 28 Pasin Manurangsi. On approximating projection games. Master’s thesis, Massachusetts Institute of Technology, January 2015.
- 29 Pasin Manurangsi and Prasad Raghavendra. A birthday repetition theorem and complexity of approximating dense csps. In *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, pages 78:1–78:15, 2017. doi:10.4230/LIPIcs.ICALP.2017.78.



- 30    Pasin Manurangsi and Daniel Reichman. The computational complexity of training relu(s). *CoRR*, abs/1810.04207, 2018. [arXiv:1810.04207](#).
- 31    Nimrod Megiddo. On the complexity of polyhedral separability. *Discrete & Computational Geometry*, 3(4):325–337, 1988.
- 32    Xingyuan Pan and Vivek Srikumar. Expressiveness of rectifier networks. In *International Conference on Machine Learning*, pages 2427–2435, 2016.
- 33    Erez Petrank. The hardness of approximation: Gap location. *Computational Complexity*, 4:133–157, 1994. doi:10.1007/BF01202286.
- 34    Tim Roughgarden. Beyond the worst-case analysis of algorithms, 2020.
- 35    Grant Schoenebeck. Linear level lasserre lower bounds for certain k-CSPs. In *FOCS*, pages 593–602, 2008. doi:10.1109/FOCS.2008.74.
- 36    Rocco A Servedio and Li-Yang Tan. What circuit classes can be learned with non-trivial savings? In *8th Innovations in Theoretical Computer Science Conference (ITCS 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.
- 37    Kirill Simonov, Fedor Fomin, Petr Golovach, and Fahad Panolan. Refined complexity of PCA with outliers. In *International Conference on Machine Learning*, pages 5818–5826, 2019.
- 38    Le Song, Santosh Vempala, John Wilmes, and Bo Xie. On the complexity of learning neural networks. In *Advances in Neural Information Processing Systems*, pages 5514–5522, 2017.
- 39    Nathan Srebro, Karthik Sridharan, and Ambuj Tewari. Smoothness, low noise and fast rates. In John D. Lafferty, Christopher K. I. Williams, John Shawe-Taylor, Richard S. Zemel, and Aron Culotta, editors, *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems 2010. Proceedings of a meeting held 6-9 December 2010, Vancouver, British Columbia, Canada*, pages 2199–2207. Curran Associates, Inc., 2010. URL: <http://papers.nips.cc/paper/3894-smoothness-low-noise-and-fast-rates>.
- 40    Madhur Tulsiani. CSP gaps and reductions in the lasserre hierarchy. In *STOC*, pages 303–312, 2009. doi:10.1145/1536414.1536457.
- 41    Santosh Vempala and John Wilmes. Polynomial convergence of gradient descent for training one-hidden-layer neural networks. In *Conference on Learning Theory*, pages 3115–3117, 2019.
- 42    Van H Vu. On the infeasibility of training neural networks with small mean-squared error. *IEEE Transactions on Information Theory*, 44(7):2892–2900, 1998.
- 43    Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint*, 2016. [arXiv:1611.03530](#).



# A Largish Sum-Of-Squares Implies Circuit Hardness and Derandomization

**Pranjal Dutta**

Chennai Mathematical Institute, India  
CSE, Indian Institute of Technology, Kanpur, India  
pranjal@cmi.ac.in

**Nitin Saxena**

Indian Institute of Technology, Kanpur, India  
nitin@cse.iitk.ac.in

**Thomas Thierauf**

Hochschule Aalen, Germany  
thomas.thierauf@uni-ulm.de

---

## Abstract

For a polynomial  $f$ , we study the *sum of squares representation (SOS)*, i.e.  $f = \sum_{i \in [s]} c_i f_i^2$ , where  $c_i$  are field elements and the  $f_i$ 's are polynomials. The size of the representation is the number of monomials that appear across the  $f_i$ 's. Its minimum is the *support-sum*  $S(f)$  of  $f$ .

For simplicity of exposition, we consider univariate  $f$ . A trivial lower bound for the support-sum of, a full-support univariate polynomial,  $f$  of degree  $d$  is  $S(f) \geq d^{0.5}$ . We show that the existence of an explicit polynomial  $f$  with support-sum just slightly larger than the trivial bound, that is,  $S(f) \geq d^{0.5+\varepsilon(d)}$ , for a sub-constant function  $\varepsilon(d) > \omega(\sqrt{\log \log d / \log d})$ , implies that  $\text{VP} \neq \text{VNP}$ . The latter is a major open problem in algebraic complexity. A further consequence is that blackbox-PIT is in SUBEXP. Note that a random polynomial fulfills the condition, as there we have  $S(f) = \Theta(d)$ .

We also consider the *sum-of-cubes representation (SOC)* of polynomials. In a similar way, we show that here, an explicit hard polynomial even implies that blackbox-PIT is in P.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Algebraic complexity theory

**Keywords and phrases** VP, VNP, hitting set, circuit, polynomial, sparsity, SOS, SOC, PIT, lower bound

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.23

**Funding** *Pranjal Dutta*: Supported by Google PhD Fellowship.

*Nitin Saxena*: Supported by DST-grant DST/SJF/MSA-01/2013-14 and N. Rama Rao Chair.

*Thomas Thierauf*: Supported by DFG-grant TH 472/5-1.

**Acknowledgements** P. D. thanks CSE, IIT Kanpur for the hospitality. Thanks to Manindra Agrawal for many useful discussions to optimize the SOS representations; to J. Maurice Rojas for several comments; to Arkadev Chattopadhyay for organizing a TIFR Seminar on this work. T. T. thanks CSE, IIT Kanpur for the hospitality.

## 1 Introduction

The sum-of-squares representation (SOS) is one of the most fundamental in number theory and algebra. Lagrange's four-squares theorem inspired generations of mathematicians [27]. Hilbert's *17th problem* asks whether a multivariate polynomial, that takes only non-negative values over the reals, can be represented as an SOS of rational functions [26]. In engineering, SOS has found many applications in approximation, optimization and control theory, see [28, 18, 19, 4]. In this work, we show a connection to central complexity questions.

Consider the following basic problem on the size of SOS-representations.



© Pranjal Dutta, Nitin Saxena, and Thomas Thierauf;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 23; pp. 23:1–23:21



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

► **Open Problem.** *Exhibit an explicit univariate polynomial  $f(x) \in \mathbb{C}[x]$  of degree  $d$  such that any SOS-representation  $f(x) = \sum_i f_i(x)^2$  requires  $\sum_i \text{sparsity}(f_i) > \omega(\sqrt{d})$ .*

Before delving into the meaning of *explicitness*, note that  $\Omega(\sqrt{d})$  is a trivial lower bound, for a polynomial of degree  $d$ , with full support (by counting monomials). Moreover, for most polynomials  $f$ , a larger lower bound of  $\Omega(d)$  holds, by a dimension argument. In other words, we ask for an explicit polynomial  $f(x)$  that has a merely largish  $\sum \wedge^2 \sum \wedge$ -formula. We show that one can bootstrap the seemingly weak hardness condition for SOS to general circuits (see Theorem 6) and to the infamous *determinant vs. permanent* question (see Corollary 14).

## 1.1 Algebraic circuits and univariate polynomials

Valiant defined the algebraic complexity classes VP and VNP based on algebraic circuits (for definitions see Section 2). They are considered as the algebraic analog of boolean classes P and NP. Separating VP from VNP is a long-standing open problem. One of the popular ways has been via depth-reduction results [3, 14, 10, 36]. It seems that showing strong lower bounds require a deeper understanding of the algebraic-combinatorial structure of circuits, which may be easier to unfold for more analytic models that appear in wider mathematics.

It is known that most of the polynomials of degree  $d$  are *hard*, i.e. they require  $\Omega(d)$  size circuits; for a self-contained proof, see [7, Theorem 4.2]<sup>1</sup>. In fact, for  $p_i$  being the  $i$ -th prime,  $\sum_{i=0}^d \sqrt{p_i} x^i$  and  $\sum_{i=0}^d 2^{2^i} x^i$ , both require circuits of size  $\Omega(d/\log d)$ , see [6, Cor.9.4] & [35]. Such polynomials can be converted to an *exponentially hard* multilinear polynomial  $f_n(\mathbf{x})$ . Unfortunately, this *strong* lower bound is insufficient to separate VP and VNP because the polynomial family is *non-explicit*—so  $f_n$  may not be in VNP. For details, see [11, 5].

Thus, the explicitness of the family plays a major role in its usefulness in algebraic complexity.

► **Definition 1 (Explicit functions).** *Let  $(f_d)_d$  be a polynomial family, where  $f_d(x)$  is of degree  $d$ . The family is explicit, if its coefficient-function is computable in time  $\text{poly} \log(d)$  and each coefficient can be at most  $\text{poly}(d)$ -bits long. The coefficient-function gets input  $(j, i, d)$  and outputs the  $j$ -th bit of the coefficient of  $x^i$  in  $f_d$ .*

Alternative versions of explicitness define the coefficient-function to be computable in  $\#P/\text{poly}$  or in the *counting hierarchy* CH, which would be good enough for our purpose (see Theorem 13).

An *explicit* candidate for the hard family is the *Pochhammer-Wilkinson* polynomial,  $f_d(x) := \prod_{i=1}^d (x - i)$ . Other explicit families, but *not* hard, are  $(x + 1)^d$  and the *Chebyshev* polynomial (that writes  $\cos d\theta$  as a function of  $\cos \theta$ ) [22]. These three are quite relevant to this work.

The interplay between proving lower bounds and derandomization is one of the central themes in complexity theory [24]. Blackbox Polynomial Identity Testing (PIT) asks for an algorithm to test the zeroness of a given algebraic circuit via mere query access. It is still an open question to design an efficient deterministic PIT algorithm. A circuit of size  $s$  can have  $\exp(s)$  many monomials. However, since a non-zero polynomial evaluated at a random point is non-zero with high probability (by the *Polynomial Identity Lemma* [25, 8, 41, 33]), one gets a randomized poly-time algorithm for PIT. For PIT refer [31, 32, 34, 23, 39].

<sup>1</sup> The size-bound in the previous such proofs usually counted only the number of nodes in the circuit, achieving square-root in the bound; we use the number of nodes and edges here.

One important direction, from hardness to derandomization, is to design deterministic PIT algorithms for small circuits assuming access to *explicit hard polynomials* [24, 13]. Most of the constructions use the concept of *hitting-set generator* (HSG), see Definition 33. Very recent work discovered that PIT is amenable to the phenomenon of *bootstrapping* (w.r.t. variables) [2, 17]. Finally, Guo et al. [9] showed: ample circuit-hardness of *constant-variate* polynomials (including univariate) implies blackbox-PIT in P.

## 1.2 Sum-of-squares model (SOS)

We want to relate variants of SOS to PIT and circuit lower bounds. Towards that, we show a connection between large SOS representation and hard polynomials; strong enough to imply  $\text{VP} \neq \text{VNP}$  and subsequently  $\text{PIT} \in \text{SUBEXP}$ . This is mainly achieved by an SOS-decomposition result for circuits via Algebraic Branching Programs (ABP) (for definition see Section 2). It expresses any  $d$ -degree polynomial  $f(\mathbf{x})$  of circuit size  $s$  as sum of squares of polynomials with degree at most  $d/2$ . We manage the top-fanin of SOS within a quasi-polynomial blow-up. Finally, we apply a careful *multi-linearization* trick to convert the hardness from the univariate SOS-model to general circuits.

► **Definition 2** (SOS and support-sum size  $S_R(f)$ ). *Let  $R$  be a ring. An  $n$ -variate polynomial  $f(\mathbf{x}) \in R[\mathbf{x}]$  is represented as a (weighted) sum-of-squares (SOS), if*

$$f = \sum_{i=1}^s c_i f_i^2, \quad (1)$$

for some top-fanin  $s$ , where  $f_i(\mathbf{x}) \in R[\mathbf{x}]$  and  $c_i \in R$ .

The size of the representation of  $f$  in (1) is the support-sum, the sum of the support size (or sparsity) of the polynomials  $f_i$ . The support-sum size of  $f$ , denoted by  $S_R(f)$ , is defined as the minimum support-sum of  $f$ .

► **Remark 3.** In real analysis, the SOS representation of a polynomial is defined without the coefficients  $c_i$ , that is, only for non-negative polynomials  $f$ . In these terms, what we define in (1) is a *weighted* SOS. However, we will skip the term “weighted” in the following.

If we consider the expression in (1) as a  $\sum \bigwedge^2 \sum \prod$ -formula, then the support-sum is the number of  $\prod$ -operations directly above the input level.

For any  $N$ -variate polynomial  $f$  of degree  $d$ . Let  $|f|_0$  denote the sparsity of  $f$ . For any field  $R = \mathbb{F}$  of characteristic  $\neq 2$ , we have

$$|f|_0^{1/2} \leq S_{\mathbb{F}}(f) \leq 2|f|_0 + 2. \quad (2)$$

The lower bound can be shown by counting monomials. The upper bound is because

$$f = (f+1)^2/4 - (f-1)^2/4. \quad (3)$$

In particular, the SOS-model is *complete* for any field of characteristic  $\neq 2$ . It can be argued by a geometric-dimension argument that for most  $N$ -variate (constant  $N \geq 1$ ) polynomials  $f$  of degree  $d$ , we have  $S_{\mathbb{F}}(f(\mathbf{x})) = \Theta(d^N)$ , as for random  $f$ ,  $|f|_0 = \Theta(d^N)$ .

We want to explore how  $S_{\mathbb{F}}(f_d)$  behaves w.r.t.  $d$ , for *explicit* families  $(f_d)_d$ , that is, the coefficient-function of the family is computable in time  $\text{poly}(\log d)$ . We call a polynomial family SOS-hard, if its support-sum is just *slightly* larger than the trivial lower bound from (2).

► **Definition 4** (SOS-hardness). *For constant  $N \geq 1$ , an explicit  $N$ -variate polynomial family  $(f_d(\mathbf{x}))_d$  is SOS-hard, if  $S_{\mathbb{F}}(f_d) = \Omega(d^{N(0.5+\varepsilon)})$ , where  $\varepsilon := \varepsilon(d) = \omega\left(\sqrt{\frac{\log \log d}{\log d}}\right)$  is a sub-constant function.*

► **Remark 5.**

1. For our purpose we could relax the explicitness condition such that the  $j$ -th bit of  $\text{coef}_{\mathbf{x}^i}(f_d)$  is computable in  $\text{poly}(2^{1/\varepsilon})$  time. This makes the family *barely explicit* w.r.t.  $d$ . In fact,  $\#P/\text{poly}$  w.r.t.  $2^{1/\varepsilon}$  works too. Eg.  $f_d = \sum_{i \in [d]} 2^{i^2} x^i$  is an easy candidate for  $N = 1$ .
2.  $\Omega(d^{N(0.5+\varepsilon)})$ , instead of  $\Omega(d^N)$ , which is the expected bound for most  $f_d$ , is a much weaker requirement. In fact, the trivial lower bound is  $S(f_d) \geq \Omega(d^{N/2})$ . Thus, we demand just a tiny improvement over the trivial bound, namely, by a factor of  $d^{N\varepsilon} = d^{o(1)}$ . For example,  $(\log d)^{\sqrt{\log d}}$  is such a function that works in  $d^\varepsilon$ .

### 1.3 Our results for SOS

Algebraic circuits are quite well-structured, for eg. , there is a famous depth- $O(\log d)$  reduction result [38, 34, 29]. Its proof methods implicitly establish (see Lemma 28) that an  $n$ -variate, degree  $d$  polynomial  $f(\mathbf{x})$ , computed by a circuit of size  $s$ , can be rewritten as

$$f(\mathbf{x}) = \sum_{i=1}^{O(sd^2)} c_i f_i(\mathbf{x})^2, \quad (4)$$

for  $c_i \in \mathbb{F}$  and  $f_i \in \mathbb{F}[\mathbf{x}]$ , where each  $f_i$  has circuit size at most  $O(sd^2)$  and  $\deg(f_i) \leq 2d/3$ , for all  $i$ . Moreover, with a larger, *quasi*-polynomial blowup in the top-fanin, we bring down the degree really to  $d/2$  (via Algebraic Branching Programs (ABP)); for the details, see Section 3.1.

► **Main Lemma** (SOS Decomposition). *Let  $\mathbb{F}$  be a field of characteristic  $\neq 2$ . Let  $f(\mathbf{x})$  be an  $n$ -variate polynomial over  $\mathbb{F}$  of degree  $d$ , computed by a circuit of size  $s$ . Then there exist  $f_i \in \mathbb{F}[\mathbf{x}]$  and  $c_i \in \mathbb{F}$  such that  $f(\mathbf{x}) = \sum_{i=1}^{s'} c_i f_i(\mathbf{x})^2$ , for  $s' \leq (sd)^{O(\log d)}$  and  $\deg(f_i) \leq \lceil d/2 \rceil$ , for all  $i \in [s']$ .*

The leitmotif of this paper is the interplay between SOS-hardness and derandomization/hardness questions in algebraic complexity. Could a barely explicit and mildly hard polynomial in the SOS-model settle the VP vs. VNP question? We evince a positive answer.

► **Theorem 6** (Circuit hardness). *If there exists an SOS-hard family, then  $\text{VP} \neq \text{VNP}$ .*

► **Remark 7.**

1. Our proof-method from constant- $N$ -variate SOS-hardness to  $\text{VP} \neq \text{VNP}$  is essentially the same as the one for  $N = 1$  (eg. replace  $d$  by  $d^N$ ). So, for simplicity of exposition, from now on we will focus on univariate SOS-hardness.
2. In the *non-commutative* setting, lower bound on sum-of-squares (of multivariates) implies that Permanent is hard [12]. Our theorem can be seen as its natural analog in the commutative setting; where potential cancellations could give smaller representations.
3. Another simple candidate for SOS-hardness is  $f_d = (x+1)^d$  (though, by repeated squaring, it has circuit size  $\Theta(\log d)$ ). However, its coefficients are not  $\text{poly} \log(d)$ -time explicit. Nevertheless, from its CH-explicitness, and GRH, the theorem does hold. Similarly, for the polynomial family,  $f_d(x) = \prod_{i \in [d]} (x-i)$  and  $f_d(x) = \sum_{0 \leq i \leq d} x^i / i!$ , and Chebyshev polynomials.

4. In the theorem and Equation (1), we could restrict the degrees of  $f_i$  to be  $O(d\varepsilon \log d) = d \cdot o(\log d)$  and the top-fanin  $s = d^{o(\varepsilon)} = d^{o(1)}$ . (Also, Corollary 14 works with analogously weaker  $\varepsilon$ .) This might help in constructing polynomials with a weaker SOS-hardness notion. See Section 3.2 for more details.
5. A stronger SOS-hardness notion with *constant*  $\varepsilon$ , gives an *exponential* separation between VP and VNP. This proof has many technical differences; refer to Theorem 32 for the details.

Hardness of general circuits often leads to nontrivial *derandomization* [24, 13, 2, 9]. Our methods in Theorem 6 consequently put blackbox-PIT in SUBEXP [13, Thm. 7.7]. In fact, if  $\varepsilon$  is a constant, then it puts blackbox-PIT  $\in$  QP (*Quasi*-polynomial-time) (Theorem 32).

## 1.4 Sum-of-cubes model (SOC)

We show that a strong lower bound in the sum-of-cubes model leads to a *complete* derandomization of blackbox-PIT. We say that an  $n$ -variate polynomial  $f(\mathbf{x}) \in R[\mathbf{x}]$  over a ring  $R$  is computed as a *sum-of-cubes* (SOC), if

$$f = \sum_{i=1}^s c_i f_i^3, \quad (5)$$

for some top-fanin  $s$ , where  $f_i(\mathbf{x}) \in R[\mathbf{x}]$  and  $c_i \in R$ .

► **Definition 8** (Support-union size  $U_R(f, s)$ ). *The size of the representation of  $f$  in (5) is the size of the support-union, namely the number of distinct monomials in the representation,  $|\bigcup_{i=1}^s \text{supp}(f_i)|$ , where  $\text{supp}(f_i)$  denotes the set of monomials with a nonzero coefficient in the polynomial  $f_i(\mathbf{x})$ . The support-union size of  $f$  with respect to  $s$ , denoted  $U_R(f, s)$ , is defined as the minimum support-union size when  $f$  is written as in (5).*

If we consider the expression in (5) as a  $\sum \wedge^3 \sum \Pi$ -circuit, then the support-union size is the number of  $\Pi$ -operations directly above the input level (unlike  $\sum \wedge^2 \sum \Pi$ -formula in Definition (2)).

The two measures—support-union and support-sum—are largely incomparable, since  $U(\cdot)$  has the extra argument  $s$ . Still one can show:  $S_{\mathbb{F}}(f) \geq \min_s (U_{\mathbb{F}}(f, 4s) - 1)$  (Lemma 26).

For any polynomial  $f$  of sparsity  $|f|_0$ , we have

$$|f|_0^{1/3} \leq U_{\mathbb{F}}(f, s) \leq |f|_0 + 1, \quad (6)$$

where the upper bound is for  $s \geq 3$  and for fields  $R = \mathbb{F}$  of characteristic  $\neq 2, 3$ . The lower bound can be shown by counting monomials. The upper bound is because

$$f = (f+2)^3/24 + (f-2)^3/24 - f^3/12. \quad (7)$$

Hence, the SOC-model is *complete* for any field of characteristic  $\neq 2, 3$ .

For simplicity, fix #variables  $N = 1$ . Here are two more examples (that we know of) for the trade-off between  $s$  and the measure  $U_{\mathbb{F}}(f, s)$ , for any  $f$ .

### ► Example 9.

1. For small  $s = \Theta(d^{1/2})$ , we have  $U_{\mathbb{F}}(f, s) = O(d^{1/2})$  (Corollary 23).
2. For large  $s = \Omega(d^{2/3})$ , we have  $U_{\mathbb{F}}(f, s) = \Theta(d^{1/3})$  (Theorem 24).

However, it is unclear whether, over  $\mathbb{F} = \mathbb{Q}$ , for a very small fanin  $s$ , support-union  $= o(d)$  exists. This trade-off between the measure  $U$  and the top-fanin  $s$  in the above examples, motivated us to define hardness in the SOC-model as follows.

► **Definition 10** (SOC-hardness). A  $\text{poly}(d)$ -time explicit univariate polynomial family  $(f_d)_d$  is SOC-hard, if there exists a positive constant  $\varepsilon' < 1/2$  such that  $U_{\mathbb{F}}(f_d, d^{\varepsilon'}) = \Omega(d)$ .

## 1.5 Our results for SOC

Though technically incomparable, the SOC-hardness feels stronger than SOS-hardness (for  $N = 1$ ); indeed it can be used to prove a connection like Theorem 6. Now, we show an even stronger consequence – a complete derandomization of blackbox-PIT.

► **Theorem 11** (Derandomization). *If there is an SOC-hard family, then blackbox-PIT  $\in \mathsf{P}$ .*

► **Remark 12.**

1. Older results too lead to various conditional derandomizations. E.g. *multi*-variate hard polynomials lead to blackbox-PIT  $\in \mathsf{QP}$  (*quasipoly-time*) [13, 2]. Recently, [9] showed that the *circuit* hardness of a constant-variate polynomial family yields blackbox-PIT  $\in \mathsf{P}$  (Theorem 34). Our hardness assumption is merely in the SOC-model. In fact, SOC is the *first* restricted model where hardness implies *complete* derandomization.
2. For Theorem 11, we could restrict the degrees of  $f_i$ , to be  $O(d)$ . See Section 3.3, Remark 3.3.

## 1.6 Basic arguments

There have been a series of works that connect the hardness in restricted univariate (resp. constant-variate) models to VP vs. VNP and the PIT problem. This work is more about remodeling the major questions in the *simplest* format possible. We show how to transfer the hardness of a (univariate) polynomial family in the SOS, resp. SOC-model, to a hard (multivariate) polynomial family in the circuit-model. To do so, we adapt the existing powerful techniques to our setting. Intuitively, one would expect that the analytic nature of SOS and SOC (over  $\mathbb{R}$  or  $\mathbb{C}$ ) makes it easier to prove hardness in these models than for general circuits. In any case, we show that this would suffice to solve central questions in algebraic complexity.

The gap between the SOS-model and general circuits is mainly bridged by a decomposition lemma (Main Lemma) which emerges via ABPs. Frontiers based depth-reduction [38] implicitly shows that any polynomial  $f(\mathbf{x})$  of degree  $d$ , computed by a homogeneous circuit of size  $s$ , can be decomposed as  $f(\mathbf{x}) = \sum_{i=1}^s f_{i1} \cdot f_{i2}$ , where  $\deg(f_{ij}) \leq 2d/3$  and  $\text{size}(f_{ij}) \leq O(s)$ ; for a proof see Lemma 28. However, such proof strategies can never give intermediate polynomials of degree *exactly*  $d/2$ , simply because degree  $\approx d/2$  polynomial may not even *exist* in the computation tree, and thus, frontiers at appropriate layers do not really help. However, in the case of *homogeneous* ABPs, the intermediate degrees increase gradually, as the labels are *linear* forms. In particular, a layer of vertices computing degree *exactly*  $d/2$  exists. By cutting the ABP, say, of width  $w$ , at the  $d/2$ -th layer, we get  $f = (f_1, \dots, f_w)^T \cdot (f'_1, \dots, f'_w) = \sum_{i=1}^w f_i \cdot f'_i$ . This directly gives an SOS-form of top-fanin at most  $2w$ . The conversion from a homogeneous circuit to a homogeneous ABP is pretty straight-forward in the literature. Use log-depth-reduction [38] and induct on the depth to conclude that  $s^{O(\log d)}$ -size ABP exists. Finally, homogenize the ABP with a polynomial blowup in size. (See [16, Lem.15] or [29].)

**Proof idea of Theorem 6.** The main idea in Theorem 6 is to lift the hardness of  $f = f_d$  in the SOS-model to a multivariate polynomial, which we prove to be super-polynomially hard in the general circuit model (implying  $\notin \mathsf{VP}$ ) and explicit (implying  $\in \mathsf{VNP}$ ). Usually, to convert a univariate polynomial to multivariate, (inverse) Kronecker type substitution is used; here we *do not* use the Kronecker due to a technical barrier and the reason will



be addressed in the next paragraph. Instead, we use a *multilinear* map  $\phi$  that sends  $x^i$  to  $\phi(x^i) := \prod_{j \in [n], \ell \in [0 \dots k-1]} y_{j,\ell}$ , where  $\ell \cdot k^{j-1}$  contributes to the  $\text{base}_k(i)$ -representation in the  $j$ -th position;  $n$  and  $k$  are both functions of  $d$  to be fixed. Consider, by linear extension,  $\phi(f) =: P_{n,k}$ . By construction  $P_{n,k}$  is a  $kn$  variate  $n$  degree multilinear polynomial. With appropriate parameter fixing, we show that  $\text{size}(P_{n,k}) = (kn)^{\omega(1)}$ . The proof goes via contradiction. If the size is smaller, then using Main Lemma, we get  $P_{n,k}$  as sum of  $d^{o(\varepsilon)}$ -many  $Q_i^2$ 's; where the intermediate polynomial  $Q_i$  ( $kn$ -variate) has degree at most  $n/2$ . Thus, a naive upper bound on the support-sum (after proper parameter fixing) is  $d^{o(\varepsilon)} \cdot \binom{kn+n/2}{n/2} < d^{o(\varepsilon)} \cdot d^{1/2+\varepsilon/2} = o(d^{1/2+\varepsilon})$ , a contradiction to the SOS-hardness!

Here we remark that Kronecker type substitution *does not* give the desired result. It basically maps a monomial  $x^e$  to  $\mathbf{x}^e$ , where  $\mathbf{e} := \text{base}_{(n+1)}(e)$  for some  $n$ ; then  $n$  is the individual-degree in the image, and  $(n+1)^k \geq d+1 > n^k$ . However, this map converts  $f$  to be a  $k$ -variate, individual-degree  $n$  polynomial family and the naive binomial upper-bound on the number of terms would be  $\binom{k+kn/2}{k} > (n+1)^k > d$ ; which is useless. (Here we use  $kn/2$  as the degree of  $P_{n,k}$  is  $kn$  while the degree of the intermediate polynomial halves.) Thus, the multi-linearization trick, along with the SOS decomposition lemma via ABPs, are indispensable in our proof.

**Proof idea of Theorem 11.** The proof of Theorem 11 works very differently than that of Theorem 6. As its goal is to devise an amply hard polynomial with a *constant* number of variables only; it limits our tricks quite a bit.<sup>2</sup> It uses (inverse) Kronecker map to construct  $P_{n,k}$  from  $f = f_d$ , a constant- $k$ -variate, individual-degree  $n$  polynomial. We show this polynomial to be  $s = n^{\Omega(1)}$  hard. Recall that an explicit constant-variate *circuit*-hard polynomial can be used as an efficient hitting-set generator; showing blackbox-PIT  $\in \mathsf{P}$  [9]. The hardness result organically comes from a *SOC decomposition lemma* (Lemma 16); using a “constant-boosting” of frontier-based Lemma 28 and a “greedy clustering”. Basically, we show that any homogeneous polynomial  $P(\mathbf{x})$  of degree  $d$ , computed by a homogeneous circuit of size  $s'$ , can be written as  $P(\mathbf{x}) = \sum_{i=1}^{\text{poly}(s')} c_i \cdot Q_i(\mathbf{x})^3$ , where  $\deg(Q_i) \leq 4d/11$ .<sup>3</sup> Applying this to each homogeneous part of  $P_{n,k}$ , and then Kronecker substitution would show (with proper parameter fixing) that  $U_{\mathbb{F}}(f) \leq |\bigcup_{i=1}^{\text{poly}(s,n)} \text{supp}(Q_i)| \leq \binom{k+4kn/11}{k} < c \cdot d$ , for *any* positive constant  $c$ . We use Eqn.(8) to bound the binomial and reach a contradiction. The constant  $4/11$  is nothing special; any constant in  $(1/3, 1/e)$  would work.

Here, we remark that [9] works for constant  $k$ . Thus, any naive upper bound on the support-union size (under the optimal decomposition) would give  $\binom{k+kn/3}{k} = \Theta(d)$ . Hence, the strongest demand of  $\Omega(d)$  is required.

## 2 Preliminaries

**Basic notation.** We work with  $\mathbb{F} = \mathbb{Q}, \mathbb{Q}_p$ , or their fixed extensions. Our results hold also for large characteristic (required for Thm. 11 using [9], and Thm 13 & Lemma 19).

Let  $[n] = \{1, \dots, n\}$ . For  $i \in \mathbb{N}$  and  $b \geq 2$ , we denote by  $\text{base}_b(i)$  the unique  $k$ -tuple  $(i_1, \dots, i_k)$  such that  $i =: \sum_{j=1}^k i_j \cdot b^{j-1}$ .

<sup>2</sup> Eg. the failure analysis above with  $\binom{k+kn/2}{k}$  is also partly the reason why SOS can't give complete PIT.

<sup>3</sup> We cannot use such a decomposition lemma using ABPs, as the *super*-polynomial blowup in the fanin, owing to the larger degree ( $\approx d^{1/k}$ ), would fail to prove the desired circuit-hardness of the resulting polynomial family.



For binomial coefficients, we use an easy bound based on the  $e^k$ -series [40], for  $1 \leq k \leq n$ ,

$$\left(\frac{n}{k}\right)^k \leq \binom{n}{k} \leq \left(\frac{en}{k}\right)^k. \quad (8)$$

**Polynomials.** For  $p \in \mathbb{F}[\mathbf{x}]$ , where  $\mathbf{x} = (x_1, \dots, x_m)$ , for some  $m \geq 1$ , the *support* of  $p$ , denoted by  $\text{supp}(p)$ , is the set of nonzero monomials in  $p$ . *Sparsity* or *support size* of  $p$  is  $|p|_0 := |\text{supp}(p)|$ . By  $\text{coef}(p)$  we denote the *coefficient vector* of  $p$  (in some fixed order).

For an exponent vector  $\mathbf{e} = (e_1, \dots, e_k)$ , we use  $\mathbf{x}^{\mathbf{e}}$  to denote the monomial  $x_1^{e_1} \dots x_k^{e_k}$ .

**Algebraic circuits.** An *algebraic circuit* over a field  $\mathbb{F}$  is a layered directed acyclic graph that uses field operations  $\{+, \times\}$  and computes a polynomial. It can be thought of as an algebraic analog of boolean circuits. The leaf nodes are labeled with the input variables  $x_1, \dots, x_n$  and constants from  $\mathbb{F}$ . Other nodes are labeled as addition and multiplication *gates*. The root node outputs the polynomial computed by the circuit.

Complexity parameters of a circuit are: **1)** the *size*, i.e. number of edges and nodes, **2)** the *depth*, i.e. number of layers, **3)** the *fan-in*, i.e. maximum number of inputs to a node, (resp. the *fan-out*, i.e. maximum number of outputs of a node).

When the graph is in fact a tree, i.e., the fan-out is 1, we call the circuit an *algebraic formula*.

For a polynomial  $f$ , the size of the smallest circuit computing  $f$  is denoted by  $\text{size}(f)$ , it is the *algebraic circuit complexity* of  $f$ . By  $\mathcal{C}(n, D, s)$ , we denote the set of circuits  $C$  that compute  $n$ -variate polynomials of degree  $D$  such that  $\text{size}(C) \leq s$ .

In *complexity classes*, we specify an upper bound on these parameters. Valiant's class **VP** contains the families of  $n$ -variate polynomials of degree  $\text{poly}(n)$  over  $\mathbb{F}$ , computed by circuits of  $\text{poly}(n)$ -size. The class **VNP** can be seen as a non-deterministic analog of the class **VP**. A family of  $n$ -variate polynomials  $(f_n)_n$  over  $\mathbb{F}$  is in **VNP** if there exists a family of polynomials  $(g_n)_n$  in **VP** such that for every  $\mathbf{x} = (x_1, \dots, x_n)$  one can write  $f_n(\mathbf{x}) = \sum_{w \in \{0,1\}^{t(n)}} g_n(\mathbf{x}, w)$ , for some polynomial  $t(n)$  which is called the *witness size*. It is straightforward to see that  $\text{VP} \subseteq \text{VNP}$  and *conjectured* to be different (Valiant's Hypothesis [37]). For more details see [20, 34, 6]. Unless specified particularly, we consider the field  $\mathbb{F} = \mathbb{Q}$  (resp. a finite field with large characteristic).

Valiant [37] showed a *sufficient* condition for a polynomial family  $(f_n(\mathbf{x}))_n$  to be in **VNP**. We use a slightly modified version of the criterion and formulate it only for multi-linear polynomials. For a proof see Appendix D.

► **Theorem 13** (Valiant's VNP criterion, [37]). *Let  $f_n(\mathbf{x}) = \sum_{\mathbf{e} \in \{0,1\}^n} c_n(\mathbf{e}) \mathbf{x}^{\mathbf{e}}$  be a polynomial family such that the coefficients  $c_n(\mathbf{e})$  have length  $\leq 2^n$  in binary. Let  $c_{n,j}(\mathbf{e})$  be the  $j$ -th bit of  $c_n(\mathbf{e})$ . Then*

$$c_{n,j}(\mathbf{e}) \in \#P/\text{poly} \implies f_n \in \text{VNP}.$$

**Algebraic branching programs (ABP).** An *algebraic branching program (ABP)* in variables  $\mathbf{x}$  over a field  $\mathbb{F}$  is a directed acyclic graph with a *starting vertex*  $s$  with in-degree zero, an *end vertex*  $t$  with out-degree zero. The edge between any two vertices is labeled by affine form  $a_1x_1 + \dots + a_nx_n + c \in \mathbb{F}[\mathbf{x}]$ , where  $a_i, c \in \mathbb{F}$ .

The *weight of a path* in an ABP is the product of labels of the edges in the path. The *polynomial computed at a vertex*  $v$  is the sum of weights of all paths from the starting vertex  $s$  to  $v$ . The *polynomial computed by the ABP* is the polynomial computed at the end vertex  $t$ .

The polynomial computed by an ABP can be written as a matrix product  $U^T(\prod_i M_i)V$ , where  $U, V \in \mathbb{F}^{w \times 1}$  and  $M_i \in \mathbb{F}[\mathbf{x}]^{w \times w}$  with entries being affine linear forms. The parameter  $w$  is called the *width* of the ABP. The class **VBP** contains the families of polynomials computed by ABPs of size  $\text{poly}(n)$ . This implies that the degree is  $\text{poly}(n)$  too.

An ABP is a very restricted circuit, but still being able to compute determinants [21].

We say that the ABP is *homogeneous*, if the polynomial computed at every vertex is a homogeneous polynomial. It is known that for an ABP  $S$  of size  $s$  computing a homogeneous polynomial  $f$ , there is an equivalent homogeneous ABP  $A'$  of size  $\text{poly}(s)$ , where each edge-label is a *linear form*  $a_1x_1 + \dots + a_nx_n$ . Moreover, when  $f$  has degree  $D$ , then  $A'$  has  $D + 1$  layers and each vertex in the  $i$ -th layer computes a homogeneous polynomial of degree  $= i$  (see [16, Lem.15] or [29]).

Here, we also remark that each homogeneous part of a degree  $d$  polynomial  $f(\mathbf{x})$ , computed by  $s$ -size circuit, can also be computed by a *homogeneous* circuit of size  $O(sd^2)$ ; see [34, 29].

### 3 Proof of the main results

#### 3.1 SOS decomposition of circuits: Proof of Main Lemma

**Proof of Main Lemma.** Let  $C$  be a circuit of size  $s$  computing  $f(\mathbf{x})$ . W.l.o.g.,  $f(\mathbf{x})$  is a homogeneous polynomial (as later we will apply to every homogeneous component of  $f$ ). Using the log-depth reduction of [38], there is a homogeneous circuit  $C'$  of depth  $\log d$  and size  $\text{poly}(s)$  that computes  $F$ .

Now we convert the circuit  $C'$  to a *layered* ABP  $A$  as follows: first, convert the circuit  $C'$  to a formula  $F$ . By induction on the depth of the circuit one can show that  $F$  has size  $s^{O(\log d)}$ . Secondly, we convert  $F$  to an ABP  $A$ . It is well known that for any formula of size  $t$ , there exists an ABP of size at most  $t + 1$ , computing the same polynomial, for details see [30, Lemma 2.14]. Thus, the ABP  $A$  computing  $f$  has size at most  $s^{O(\log d)}$ .

Further, we *homogenize* the ABP  $A$  as explained at the end of the preliminary section. Let  $A'$  be the homogenized ABP computing  $f$ . Its size is  $s' := \text{poly}(s^{O(\log d)}) = s^{O(\log d)}$ .

Finally, cut ABP  $A'$  in half, at the  $\lceil d/2 \rceil$ -th layer, to get:  $f = (f_1, \dots, f_{s'})^T \cdot (f'_1, \dots, f'_{s'})$  where, degree of each  $f_i, f'_i$  is at most  $\lceil d/2 \rceil$ . This can be easily rewritten as SOS by Equation (3). The top-fanin of SOS is at most  $2s'$ .

For a non-homogeneous polynomial  $f(\mathbf{x})$ , we can apply the above for each homogeneous part of  $f(\mathbf{x})$ . It is well known that each homogeneous part can be computed by a homogeneous circuit of size  $O(sd^2)$ . Thus, for non-homogeneous polynomials,  $s$  can be replaced by  $O(sd^2)$ ; hence the top-fanin of SOS is  $(sd^2)^{O(\log d)} = (sd)^{O(\log d)}$ .  $\blacktriangleleft$

#### 3.2 SOS-hardness to VP $\neq$ VNP: Proof of Theorem 6

**Proof of Theorem 6.** We will construct an explicit (multivariate) polynomial family, using SOS-hard univariate  $f_d$ , which is not in VP, but is in VNP. This would imply that VP  $\neq$  VNP.

**Construction.** We will construct  $(P_{n,k})_k$  from  $f_d$ , where  $P_{n,k}$  is a multilinear degree- $n$  and  $kn$ -variate polynomial, for  $n = n(d)$  and  $k = k(d)$ <sup>4</sup>. We will specify  $k$  and  $n$  in the course of the proof. The basic relation between  $d, n$  and  $k$  is that  $k^n \geq d + 1 > (k - 1)^n$ . Introduce  $kn$  many new variables  $y_{j,\ell}$ , where  $1 \leq j \leq n$  and  $0 \leq \ell \leq k - 1$ . Let  $\phi_{n,k}$  be the map,

$$\phi_{n,k} : x^i \mapsto \prod_{j=1}^n y_{j,i_j}, \text{ where } i =: \sum_{j=1}^n i_j \cdot k^{j-1}, \quad 0 \leq i_j \leq k - 1.$$

<sup>4</sup> In this section think of  $n$  as a *tiny* function of  $k$ . Thus indexing the family over  $k$  suffices.

## 23:10 Largish SOS Implies Circuit Hardness

Note: for  $i \in [0, d]$ ,  $\phi_{n,k}$  maps  $x^i$  uniquely to a multilinear monomial of degree  $n$ . By linear extension, define  $\phi_{n,k}(f_d) =: P_{n,k}$ . By construction,  $P_{n,k}$  is  $n$ -degree,  $kn$ -variate multilinear polynomial. Let  $\psi_{n,k}$  be the homomorphism that maps any  $n$ -degree multilinear monomial, defined on variables  $y_{j,\ell}$ , such that  $y_{j,\ell} \mapsto x^{\ell \cdot k^{j-1}}$ . Observe that,  $\psi_{n,k} \circ \phi_{n,k}(f) = f$ , for any degree  $\leq d$  polynomial  $f \in \mathbb{F}[x]$ .

**SOS-hardness  $\implies$  hardness of  $P_{n,k}$ .** Assume that family  $(f_d)$  is SOS-hard with parameter  $\varepsilon$ . We will show that  $\text{size}(P_{n,k}) \geq d^{\mu(d)} = (kn)^{\omega(1)}$  for some function  $\mu$  depending on  $\varepsilon(d)$ . We have  $\varepsilon > \omega(\sqrt{\log \log d / \log d})$  and w.l.o.g.  $\varepsilon < (\log \log d / \log d)^{1/3}$ , for large  $d$  (Note: Proving for a small  $\varepsilon$  suffices; also  $1/3$  is nothing special, any constant  $< 1/2$  in the exponent works.).

Suppose,  $\text{size}(P_{n,k}) \leq d^\mu$ , for some  $\mu(d)$ . Then, from Main Lemma, we know that  $\exists Q_i$ 's such that  $P_{n,k} = \sum_{i=1}^s c_i \cdot Q_i^2$ , where  $s \leq (d^\mu \cdot n)^{c \log n}$ , for some constant  $c$ , with  $\deg(Q_i) \leq \lceil n/2 \rceil$ . Note:  $f_d = \psi_{n,k} \circ \phi_{n,k}(f_d) = \sum_{i=1}^s c_i \cdot \psi_{n,k}(Q_i)^2$ . As  $\psi_{n,k}$  cannot increase the sparsity,  $|\psi_{n,k}(Q_i)|_0 \leq |Q_i|_0 \leq \binom{kn + \lceil n/2 \rceil}{\lceil n/2 \rceil}$ ,<sup>5</sup> for each  $i \in [s]$ . Thus, by definition  $S_{\mathbb{F}}(f_d) \leq s \cdot \binom{kn + \lceil n/2 \rceil}{\lceil n/2 \rceil}$ . The idea is to fix parameters so that  $S(f_d) < o(d^{1/2+\varepsilon})$ . We will fix  $\mu$  such that

1.  $s \leq d^{\delta_1}$  for some function  $\delta_1$ ,
2.  $\binom{kn + \lceil n/2 \rceil}{\lceil n/2 \rceil} \leq d^{\delta_2}$  for some function  $\delta_2$ ,
3.  $d^{\delta_1 + \delta_2} < o(d^{1/2+\varepsilon})$ ,
4.  $d^\mu > (kn)^{\omega(1)}$ .

Note: from conditions 1-3, if  $\text{size}(P_{n,k}) \leq d^\mu$  then  $S(f_d) < o(d^{1/2+\varepsilon})$ , contradicting the SOS-hardness. Thus, condition 4 would give super-polynomial hardness result.

**Parameter fixing.** Let  $\mu := 1/\sqrt{\log d \cdot \log \log d}$  and  $\delta_1 := c' \cdot \mu \cdot \log n$  for some  $c' > c$ . Let  $\delta_2 := 1/2 + \varepsilon/2$ . Fix  $k := \lceil 6^{1/\varepsilon} + 1 \rceil$ . This fixing of  $k$  together with  $k^n \geq d + 1 > (k-1)^n$  implies that  $n = \Theta(\varepsilon \cdot \log d)$ . We also assume  $n$  to be even for simplicity, to avoid the ceiling function.

**Bound on the binomial.** Note that it is enough to have the following chain of inequalities:

$$\binom{kn + n/2}{n/2} \leq (e + 2ek)^{n/2} \leq (6(k-1))^{n/2} \leq (k-1)^{n\delta_2} \leq d^{\delta_2}.$$

First inequality is by Eqn.(8); the second one is by the fact that  $2e < 6$ , thus for large enough  $k$ , it holds; and the last inequality follows by the assumption that  $d \geq (k-1)^n$ . For the third one, it suffices to ensure that  $(k-1)^{\delta_2 - 1/2} \geq \sqrt{6}$ . This is where we used the fact that  $\delta_2 - 1/2 = \varepsilon/2 > 0$  and thus it is enough to fix  $k-1 = \lceil 6^{1/\varepsilon} \rceil$ .

**Bound on top-fanin  $s$ .** Note that  $s \leq (d^\mu \cdot n)^{c \log n}$  from Main Lemma for some constant  $c$ . We want  $d^{c' \cdot \mu \cdot \log n} = d^{\delta_1} \geq (d^\mu \cdot n)^{c \log n}$ . It suffices to show that  $d^{(c'-c) \cdot \mu} \geq n^c$ . It is fairly straightforward to verify that with our parameters fixing of  $\mu \log d = \sqrt{\log d / \log \log d}$ , and  $\log n \leq O(\log \log d)$ , the above inequality holds for large enough  $d$ .

**Checking  $d^{\delta_1 + \delta_2} = o(d^{1/2+\varepsilon})$ .** Note:  $\log n = O(\log \log d)$  and thus  $\delta_1 = O(\sqrt{\log \log d / \log d}) = o(\varepsilon)$ . Hence,  $\delta_1 + \delta_2 = o(\varepsilon) + 1/2 + \varepsilon/2 < 1/2 + \varepsilon$ ; since  $d^\varepsilon \rightarrow \infty$  as  $d \rightarrow \infty$ , the conclusion follows.

<sup>5</sup> Any  $n$  variate degree  $d$  polynomial can have sparsity at most  $\binom{n+d}{d}$ .

**Checking  $d^\mu = (kn)^{\omega(1)}$ .** Note that  $d^\mu = (kn)^{\omega(1)} \iff \mu = \omega(1) \cdot \log(kn)/\log d \iff \mu \cdot \log d = \omega(\log(kn))$ . It is clear that,  $\log(kn) = \log k + \log n \leq O(1/\varepsilon)$  for large enough  $n$  (or equivalently  $d$ ), as  $\log n = O(\log \log d) = o(1/\varepsilon)$  and  $\log k = \log \lceil 6^{1/\varepsilon} + 1 \rceil = O(1/\varepsilon)$ .

Also, note that  $\mu \cdot \log d = \sqrt{\log d / \log \log d} = \omega(1/\varepsilon) = \omega(\log(kn))$ .

Finally, all the conditions 1-4 are met with the appropriate fixing of parameters as shown above. Thus, we deduce  $\text{size}(P_{n,k}) \geq d^\mu = (kn)^{\omega(1)}$ , i.e.  $P_{n,k}$  requires *super*-polynomial size circuit. Therefore,  $(P_{n,k})_k \notin \text{VP}$ .

**Explicitness.** We will show that  $P_{n,k}$  is explicit, i.e.  $(P_{n,k})_k \in \text{VNP}$ . By construction,  $P_{n,k}$  is a  $kn$  variate, individual degree  $n$  multilinear polynomial, so we can write it as

$$P_{n,k} = \sum_{\mathbf{e} \in \{0,1\}^{kn}} \phi(\mathbf{e}) \cdot \mathbf{y}^{\mathbf{e}}.$$

Here  $\mathbf{y}$  denotes the  $kn$  variables  $y_{j,\ell}$  where  $1 \leq j \leq n$  and  $0 \leq \ell \leq k-1$  and  $\mathbf{e}$  denotes the exponent-vector. As each  $x^{\mathbf{e}}$  in  $\text{supp}(f_d)$  maps to a monomial  $\mathbf{y}^{\mathbf{e}}$  uniquely; given  $\mathbf{e}$ , one can easily compute  $e := \sum_{j=1}^n e_j \cdot k^{j-1}$  and thus  $\phi(\mathbf{e}) = \text{coef}_{x^{\mathbf{e}}}(f_d)$ . By the explicitness hypothesis, any bit of  $\phi(\mathbf{e})$  is computable in  $\text{poly}(\log d) < \text{poly}(2^{1/\varepsilon}) = \text{poly}(kn)$  time. Using Theorem 13, it is clear that  $(P_{n,k})_k \in \text{VNP}$ , by a wide margin.

So,  $(P_{n,k})_k \in \text{VNP}$  and SOS-hardness imply  $(P_{n,k})_k \notin \text{VP}$ . This proves Theorem 6.  $\blacktriangleleft$

► **Corollary 14** (Determinant vs Permanent). *SOS-hardness weakened with  $\varepsilon > \omega(1/\sqrt{\log d})$  (a smaller  $\varepsilon$  than the original) already implies  $\text{VBP} \neq \text{VNP}$ .*

**Proof Sketch.** The log-factor in the exponent is avoidable in the Main Lemma, if the initial polynomial is already an ABP of size  $s$  (instead of a circuit). In the above proof, we could then fix  $\delta_1 := c'\mu$ . This would remove the extra “ $\log n = \log \log d$ ” factors from the calculations.  $\blacktriangleleft$

► **Remark 15.**

1. We showed an explicit super-polynomially hard family  $(P_{n,k})_k$ . The result of [13, Theorem 7.7] then implies  $\text{PIT} \in \text{SUBEXP}$ .
2. If the given  $\varepsilon$  was a constant, say 0.001; then a very different parameters setting ( $k = O(1)$  and  $n = O(\log d)$ ) gives a *sub-exponential* hard polynomial family  $(P_{n,k})_n$  of size  $> 2^{\Omega(\log d / \log \log d)}$ . This happens because of the *super-polynomial* blowup in the size while converting a circuit to an ABP in Main Lemma. However, a repeated boosting of [38] type lemma (Lemma 31) gives a decomposition with intermediate polynomials having degree *close* to  $d/2$ . Finally this gives a truly *exponential* hard family  $(P_{k,n})_n$ ; for details see Theorem 32. Thus, [13] gives  $\text{PIT} \in \text{QP}$ , when  $\varepsilon$  is a constant.

Here, we also remark that “halving” the degree with  $\log d$  exponent in the top-fanin gives *better* result than “close” to halving because finally the contribution of the exponent is *quite small* in our application (and in fact absent in case of Corollary 14). However, for constant  $\varepsilon$ , the scenario changes as mentioned above.

3. As  $\deg(Q_i) \leq n/2$ , we have  $\deg(\psi_{n,k}(Q_i)) \leq n/2 \cdot (k-1) \cdot k^{n-1} < n \cdot k^n = O(nd) = o(d \log d)$ . Here we used that  $k^n/(k-1)^n < (1 + 1/(k-1))^n < e$ , for large  $d$ . Thus, it is enough to consider the restricted-degree SOS representation and prove the conjecture.
4. One can further restrict (proof requirement-wise) the SOS top-fanin to a mere  $d^{\delta_1} = \exp(O(\sqrt{\log d \cdot \log \log d}))$  which is extremely small compared to  $d$  (in fact,  $d^{\delta_1} = d^{o(\varepsilon)}$ ).

### 3.3 SOC-hardness to blackbox-PIT $\in$ P: Proof of Theorem 11

**Proof of Theorem 11.** The idea is to convert the SOC-hard polynomial  $f_d(x)$  to a *constant-k-variate individual-degree-n* polynomial family  $(P_{n,k})_n$  which is “mildly” *hard*. Later, using [9], we will conclude that blackbox-PIT  $\in$  P. The following lemma is the *crucial* ingredient to connect general circuits to an SOC representation.

► **Lemma 16** (SOC decomposition). *Let  $\mathbb{F}$  be a field of characteristic  $\neq 2, 3$ . Let  $f(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$  be an  $n$ -variate, degree- $d$  polynomial, computed by a circuit of size  $s$ . Then there exist polynomials  $f_i \in \mathbb{F}[x]$  and  $c_i \in \mathbb{F}$  such that  $f(\mathbf{x}) = \sum_{i=1}^{s'} c_i \cdot f_i^3$ , for some top-fanin  $s' \leq \text{poly}(s, d)$ ; achieving  $\deg(f_i) < 4d/11$ , for all  $i \in [s']$ .*

**Proof of Lemma 16.** We will first show this for homogeneous polynomials, and then apply it to each homogeneous part of a general  $f(\mathbf{x})$ . Assume that, circuit of  $f(\mathbf{x})$  is homogeneous. Lemma 28 establishes that  $f(\mathbf{x})$  can be decomposed as  $\sum_{i=1}^s \tilde{f}_{i1} \cdot \tilde{f}_{i2}$ , where  $\tilde{f}_{ij}$  has circuits of size  $O(s)$  and  $\deg(\tilde{f}_{ij}) \leq 2d/3$ , with  $\deg(\tilde{f}_{i1}) + \deg(\tilde{f}_{i2}) = d$ .

Choose a constant  $m$  such that  $(2/3)^m < 4/11 - 1/3 = 1/33$  ( $m := 9$  suffices). Apply Lemma 28  $m$  times, recursively on each successive circuit  $\tilde{f}_{ij}$ . As  $m$  is constant, it is easy to conclude that  $f(\mathbf{x})$  can be written as

$$f(\mathbf{x}) = \sum_{i=1}^{\text{poly}(s)} g_{i,1} \cdot g_{i,2} \cdot \dots \cdot g_{i,2^m},$$

where  $\deg(g_{i,j}) \leq (2/3)^m \cdot d$ , and  $\text{size}(g_{ij}) = O(s)$ . For each product  $g_{i,1} \cdot \dots \cdot g_{i,2^m}$ , pick a  $j_1 \in [2^m]$  such that  $d/3 \leq \sum_{k=1}^{j_1} \deg(g_{i,k}) < 4d/11$ . As each  $\deg(g_{i,k})$  is less than the gap between upper and lower bounds, namely  $4d/11 - d/3$ , such  $j_1$  exists. Note that  $\sum_{k=j_1+1}^{2^m} \deg(g_{i,k}) > d - 4d/11 = 7d/11 > d/3$ . Choose a  $[2^m] \ni j_2 > j_1$  such that  $d/3 \leq \sum_{k=j_1+1}^{j_2} \deg(g_{i,k}) < 4d/11$ ; such  $j_2$  exists by a similar argument.

Define,  $f_{i1} := g_{i,1} \cdot \dots \cdot g_{i,j_1}$ ,  $f_{i2} := g_{i,j_1+1} \cdot \dots \cdot g_{i,j_2}$ , and  $f_{i3} := g_{i,j_2+1} \cdot \dots \cdot g_{i,2^m}$ . By definition,  $\deg(f_{i1}), \deg(f_{i2}) \in [d/3, 4d/11]$ . As,  $\deg(f_{i1}) + \deg(f_{i2}) + \deg(f_{i3}) = \sum_{k \in [2^m]} \deg(g_{i,k}) = d \implies \deg(f_{i3}) \leq d/3 < 4d/11$ . As each  $g_{i,j}$  has a homogeneous circuit of size  $O(s)$ , so does  $f_{ij}$ . Hence,  $f(\mathbf{x}) = \sum_{i=1}^{\text{poly}(s)} f_{i1} \cdot f_{i2} \cdot f_{i3}$ . Use the identity

$$24 \cdot a \cdot b \cdot c = (a + b + c)^3 - (a - b + c)^3 - (a + b - c)^3 + (a - b - c)^3, \quad (9)$$

to write each  $f_{i1} \cdot f_{i2} \cdot f_{i3}$  as sum of four cubes. Relabeling yields  $f(\mathbf{x}) = \sum_{i=1}^{\text{poly}(s)} c_i \cdot f_i^3$ . As each  $f_i$  is a linear combination of  $f_{jk}$ 's, the degree does not change and the size is still  $O(s)$ .

It is well known that each homogeneous part can be computed by a homogeneous circuit of size  $O(sd^2)$ . Thus, for non-homogeneous polynomials,  $s$  can be replaced by  $O(sd^2)$  and the conclusion follows. ◀

Let  $k$  be a constant (to be fixed later) and  $\mathbf{x} := (x_1, \dots, x_k)$ . For all large enough  $n \in \mathbb{N}$ , define  $d := d(n) := (n+1)^k - 1$ . Let  $P_{n,k}$  be a  $k$ -variate polynomial of individual degree at most  $n$  such that after the Kronecker substitution,  $P_{n,k}(x, x^{n+1}, \dots, x^{(n+1)^{k-1}}) := f_d$ . It is easy to construct  $P_{n,k}$  from a given  $d$ ; just convert every  $x^e \in \text{supp}(f_d)$  to  $x_1^{e_1} \cdot \dots \cdot x_k^{e_k}$ , where  $e = \sum_{i=1}^k e_i \cdot (n+1)^{i-1}$  and  $0 \leq e_i \leq n$ .

By the explicitness of  $f_d$ ,  $(P_{n,k})_n$  is a very explicit polynomial family; its coefficient-vector  $\text{coef}(P_{n,k})$  can be computed in  $\text{poly}(d) = \text{poly}(n)$  time.

Next, we will show the hardness of the polynomial family  $(P_{n,k})_n$ . The SOC-hardness implies that there exists a constant  $\delta$  such that  $U(f_d, d^{\varepsilon'}) \geq \delta \cdot d$ , for all large enough  $d$ . Also, let  $c$  be the constant such that  $s' =: (sd)^c$  in Lemma 16. Let  $\mu := 2/(\varepsilon'/c - 1/k)$ , and later we will choose  $k > c/\varepsilon'$ .

▷ **Claim 17 (Hardness of  $P_{n,k}$ ).**  $\text{size}(P_{n,k}) > d^{1/\mu}$ , for all large enough  $n$ .

Assume to the contrary, that there exists an infinite subset  $J \subset \mathbb{N}$  such that  $\text{size}(P_{n,k}) \leq d^{1/\mu}$ , for all  $n \in J$ . We will show that family  $(f_d)$  is not SOC-hard over an infinite subset  $J' := \{d : n \in J\} \subseteq \mathbb{N}$ , which is a contradiction.

Let  $C$  be a circuit of size  $\leq d^{1/\mu}$  that computes  $P_{n,k}$ , for some  $n$ . Then, using Lemma 16, we know that there exist  $Q_i \in \mathbb{F}[x]$ , of degree at most  $4 \cdot \deg(P_{n,k})/11 \leq 4kn/11$ , such that  $P_{n,k} = \sum_{i=1}^{s_0} c_i \cdot Q_i^3$ , where  $s_0 \leq (d^{1/\mu} \cdot kn)^c$ . Apply the Kronecker map  $x_i \mapsto x^{(n+1)^{i-1}}$  on both sides yields  $f_d = \sum_{i=1}^{s_0} c_i \cdot \tilde{Q}_i^3$ , where  $\tilde{Q}_i := Q_i(x, x^{n+1}, \dots, x^{(n+1)^{k-1}})$ . Since Kronecker substitution cannot increase the support size,  $|\bigcup_i \text{supp}(\tilde{Q}_i)| \leq |\bigcup_i \text{supp}(Q_i)| \leq \binom{k+4kn/11}{k} =: s_1$ . Thus,  $U_{\mathbb{F}}(f_d, s_0) \leq s_1$ .

We want to show that  $s_0 < d^{\varepsilon'}$  and  $s_1 < \delta \cdot d$ , for all large enough  $n$ . Then, we have  $U_{\mathbb{F}}(f_d, d^{\varepsilon'}) < \delta \cdot d$ , for all large  $d \in J' \subset \mathbb{N}$ ; which contradicts the SOC-hardness of  $f_d$ .

**Bound on  $s_0$ .** We have for large enough  $n$  (and thus  $d$ ),

$$s_0 \leq (d^{1/\mu} \cdot k \cdot n)^c < d^{c/\mu} \cdot k^c \cdot d^{c/k} = k^c \cdot d^{c/\mu + c/k} < d^{\varepsilon'}.$$

We used that  $d = (n+1)^k - 1 > n^k$  for large  $n$ , and  $\mu > 1/(\varepsilon'/c - 1/k) \iff 1/\mu + 1/k < \varepsilon'/c$ .

**Bound on  $s_1$ .** By Eqn.(8), we have

$$s_1 = \binom{k + 4nk/11}{k} \leq (e(1 + 4n/11))^k < (10.9n/11)^k < (10.9/11)^k \cdot d.$$

As  $4e \approx 10.873$ , we used that  $e(1 + 4n/11) < (10.9/11) \cdot n$  and  $d > n^k$ , for large  $n$ .

Therefore, it suffices to show that  $(10.9/11)^k < \delta$ . Choose  $k > \log_{11/10.9}(1/\delta)$ . It suffices, from the above calculations, to pick  $k > \max(c/\varepsilon', \log_{11/10.9}(1/\delta))$ . This proves Claim 17. ◀

**From hardness to HSG.** We show that from the hardness of  $P_{n,k}$  in Claim 17, we can fulfil the assumption in Theorem 34:  $\text{size}(P_{n,k}) > s^{10k+2} \deg(P_{n,k})^3$ , for some “growing” function  $s = s(n)$ . Recall that  $\deg(P_{n,k}) \leq kn$ . We define,  $s(n) := n^{1/(10k+3)}$ . Then we have

$$s^{10k+2} (kn)^3 = n^{(10k+2)/(10k+3)} (kn)^3 = k^3 n^{4 - (1/(10k+3))} < n^4, \quad (10)$$

for large enough  $n$ . Additionally, assume that  $4 \leq k/\mu$ . Recall the fact:  $n^k < d$  for large  $n$ . So, we can continue Eqn.(10) as

$$n^4 \leq n^{k/\mu} < d^{1/\mu} < \text{size}(P_{n,k}). \quad (11)$$

Equations (10) and (11) give the desired hardness of  $P_{n,k}$ . It remains to ensure the last requirement of  $4 \leq k/\mu$ . We show below that choosing  $k \geq 9c/\varepsilon'$  suffices:

$$\mu = 2/(\varepsilon'/c - 1/k) \leq 2/(9/k - 1/k) = k/4.$$

Hence our final choice for  $k$  is:  $k \geq \max(9c/\varepsilon', \log_{11/10.9}(1/\delta))$ .

Thus, Theorem 34 gives a  $\text{poly}(s)$ -time HSG for  $\mathcal{C}(s, s, s)$ . Hence, blackbox-PIT  $\in \mathcal{P}$ . ◀

▶ **Remark 18.** Recall the proof notation. As the degree of  $Q_i$ 's is  $< 4kn/11$ , the degree of  $\tilde{Q}_i$  is  $\leq (n+1)^{k-1} \cdot 4kn/11 < 4k/11 \cdot (n+1)^k = 4k/11 \cdot (d+1) = O(d)$  ( $\because k$  is a constant). Thus, it suffices to study the representation of  $f_d$  as sum-of-cubes  $\tilde{Q}_i^3$ , where  $\deg(\tilde{Q}_i) \leq O(d)$ .



## 4 Conclusion

This work established that studying the univariate sum-of-squares representation (resp. cubes) is fruitful. Proving a *vanishingly* better lower bound than the trivial one, suffices to both derandomize and prove hardness in algebraic complexity.

Here are some immediate questions which require rigorous investigation.

1. Does existence of a SOS-hard family solve PIT completely? The current proof technique fails to reduce from cubes to squares.
2. Prove existence of a SOS-hard family for the *sum of constantly* many squares.
3. Prove existence of a SOC-hard family for a “generic” polynomial  $f$  with rational coefficients ( $\mathbb{Q}$ ). Does it fail when we move to *complex* coefficients ( $\mathbb{C}$ )?
4. Can we optimize  $\varepsilon$  in the SOS-hardness condition (& Corollary 14)? In particular, does proving an SOS lower-bound of  $\sqrt{d} \cdot \text{poly}(\log d)$ , suffice to deduce a separation between determinant and permanent (similarly VP and VNP)?

---

## References

- 1 Manindra Agrawal. Private Communication, 2020.
- 2 Manindra Agrawal, Sumanta Ghosh, and Nitin Saxena. Bootstrapping variables in algebraic circuits. *Proceedings of the National Academy of Sciences*, 116(17):8107–8118, 2019. Earlier in Symposium on Theory of Computing, 2018 (STOC’18). doi:10.1073/pnas.1901272116.
- 3 Manindra Agrawal and V Vinay. Arithmetic circuits: A chasm at depth four. In *Foundations of Computer Science, 2008. FOCS’08. IEEE 49th Annual IEEE Symposium on*, pages 67–75. IEEE, 2008. URL: <https://ieeexplore.ieee.org/document/4690941>.
- 4 Boaz Barak and Ankur Moitra. Noisy tensor completion via the Sum-of-squares Hierarchy. In *Conference on Learning Theory*, pages 417–445, 2016. URL: <http://proceedings.mlr.press/v49/barak16.pdf>.
- 5 Peter Bürgisser. *Completeness and Reduction in Algebraic Complexity Theory*, volume 7. Springer Science & Business Media, 2013. URL: <https://www.springer.com/gp/book/9783540667520>.
- 6 Peter Bürgisser, Michael Clausen, and Amin Shokrollahi. *Algebraic Complexity Theory*, volume 315. Springer Science & Business Media, 2013. URL: <https://www.springer.com/gp/book/9783540605829>.
- 7 Xi Chen, Neeraj Kayal, and Avi Wigderson. *Partial derivatives in arithmetic complexity and beyond*. Now Publishers Inc, 2011. URL: <https://www.math.ias.edu/~avi/PUBLICATIONS/ChenKaWi2011.pdf>.
- 8 Richard A. Demillo and Richard J. Lipton. A probabilistic remark on algebraic program testing. *Information Processing Letters*, 7(4):193–195, 1978. URL: <https://www.sciencedirect.com/science/article/abs/pii/0020019078900674>.
- 9 Zeyu Guo, Mrinal Kumar, Ramprasad Satharishi, and Noam Solomon. Derandomization from algebraic hardness: Treading the borders. In *60th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2019*, pages 147–157, 2019. Online version: <https://mrinalkr.bitbucket.io/papers/newprg.pdf>. doi:10.1109/FOCS.2019.00018.
- 10 Ankit Gupta, Pritish Kamath, Neeraj Kayal, and Ramprasad Satharishi. Arithmetic circuits: A chasm at depth three. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 578–587. IEEE, 2013. URL: <https://epubs.siam.org/doi/pdf/10.1137/140957123>.
- 11 Joos Heintz and Malte Sieveking. Lower bounds for polynomials with algebraic coefficients. *Theoretical Computer Science*, 11(3):321–330, 1980. URL: <https://www.sciencedirect.com/science/article/pii/0304397580900195>.



- 12 Pavel Hrubeš, Avi Wigderson, and Amir Yehudayoff. Non-commutative circuits and the sum-of-squares problem. *Journal of the American Mathematical Society*, 24(3):871–898, 2011. URL: <https://www.ams.org/journals/jams/2011-24-03/S0894-0347-2011-00694-2/S0894-0347-2011-00694-2.pdf>.
- 13 Valentine Kabanets and Russell Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Computational Complexity*, 13(1-2):1–46, 2004. doi:10.1007/s00037-004-0182-6.
- 14 Pascal Koiran. Arithmetic circuits: The chasm at depth four gets wider. *Theoretical Computer Science*, 448:56–65, 2012. doi:10.1016/j.tcs.2012.03.041.
- 15 Pascal Koiran and Sylvain Perifel. Interpolation in Valiant’s theory. *Computational Complexity*, 20(1):1–20, 2011. doi:10.1007/s00037-011-0002-8.
- 16 Mrinal Kumar. A quadratic lower bound for homogeneous algebraic branching programs. *computational complexity*, 28(3):409–435, 2019.
- 17 Mrinal Kumar, Ramprasad Saptharishi, and Anamay Tengse. Near-optimal Bootstrapping of Hitting Sets for Algebraic Circuits. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 639–646, 2019. doi:10.5555/3310435.3310475.
- 18 Jean B Lasserre. A sum of squares approximation of nonnegative polynomials. *SIAM review*, 49(4):651–669, 2007. URL: <https://epubs.siam.org/doi/abs/10.1137/070693709?journalCode=siread>.
- 19 Monique Laurent. Sums of squares, moment matrices and optimization over polynomials. In *Emerging applications of algebraic geometry*, pages 157–270. Springer, 2009. URL: <https://homepages.cwi.nl/~monique/files/moment-ima-update-new.pdf>.
- 20 Meena Mahajan. Algebraic Complexity Classes. In *Perspectives in Computational Complexity*, pages 51–75. Springer, 2014. doi:10.1007/978-3-319-05446-9\_4.
- 21 Meena Mahajan and V Vinay. Determinant: Old algorithms, new insights. *SIAM Journal on Discrete Mathematics*, 12(4):474–490, 1999.
- 22 John C Mason and David C Handscomb. *Chebyshev polynomials*. CRC press, 2002. URL: <https://books.google.co.in/books?id=g1DMBQAAQBAJ>.
- 23 Ketan D. Mulmuley. The GCT program toward the P vs. NP problem. *Commun. ACM*, 55(6):98–107, June 2012. doi:10.1145/2184319.2184341.
- 24 Noam Nisan and Avi Wigderson. Hardness vs randomness. *Journal of computer and System Sciences*, 49(2):149–167, 1994. URL: <https://www.sciencedirect.com/science/article/pii/S0022000005800431>.
- 25 Øystein Ore. Über höhere kongruenzen. *Norsk Mat. Forenings Skrifter*, 1(7):15, 1922.
- 26 Albrecht Pfister. Hilbert’s seventeenth problem and related problems on definite forms. In *Mathematical Developments Arising from Hilbert Problems, Proc. Sympos. Pure Math, XXVIII.2.AMS*, volume 28, pages 483–489, 1976. URL: <https://www.ams.org/books/pspum/028.2/>.
- 27 Srinivasa Ramanujan. On the expression of a number in the form  $ax^2 + by^2 + cz^2 + du^2$ . In *Proc. Cambridge Philos. Soc.*, volume 19, pages 11–21, 1917. URL: <http://ramanujan.sirinudi.org/Volumes/published/ram20.pdf>.
- 28 Bruce Reznick. Extremal psd forms with few terms. *Duke mathematical journal*, 45(2):363–374, 1978. URL: <https://www.math.ucdavis.edu/~deloera/MISC/LA-BIBLIO/trunk/ReznickBruce/Reznick3.pdf>.
- 29 Ramprasad Saptharishi. A survey of lower bounds in arithmetic circuit complexity. Github survey, 2019. URL: <https://github.com/dasarpmar/lowerbounds-survey/releases>.
- 30 Nitin Saurabh. Algebraic models of computation. MS Thesis, 2012. URL: [https://www.imsc.res.in/~nitin/pubs/ms\\_thesis.pdf](https://www.imsc.res.in/~nitin/pubs/ms_thesis.pdf).
- 31 Nitin Saxena. Progress on Polynomial Identity testing. *Bulletin of the EATCS*, 99:49–79, 2009. URL: <https://www.cse.iitk.ac.in/users/nitin/papers/pit-survey09.pdf>.
- 32 Nitin Saxena. Progress on Polynomial Identity Testing - II. *Perspectives in Computational Complexity*, 26:131–146, 2014. doi:10.1007/978-3-319-05446-9\_7.

- 33 J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *J. ACM*, 27(4):701–717, October 1980. doi:10.1145/322217.322225.
- 34 Amir Shpilka and Amir Yehudayoff. Arithmetic Circuits: A survey of recent results and open questions. *Foundations and Trends® in Theoretical Computer Science*, 5(3–4):207–388, 2010. URL: 10.1561/04000000039.
- 35 Volker Strassen. Polynomials with rational coefficients which are hard to compute. *SIAM Journal on Computing*, 3(2):128–149, 1974. URL: 10.1137/0203010.
- 36 Sébastien Tavenas. Improved bounds for reduction to depth 4 and depth 3. *Information and Computation*, 240:2–11, 2015. URL: <https://www.sciencedirect.com/science/article/pii/S0890540114001138>.
- 37 Leslie G Valiant. Completeness classes in algebra. In *Proceedings of the 11th Annual ACM symposium on Theory of computing*, pages 249–261. ACM, 1979. doi:10.1145/800135.804419.
- 38 Leslie G. Valiant, Sven Skyum, S. Berkowitz, and Charles Rackoff. Fast parallel computation of polynomials using few processors. *SIAM Journal of Computing*, 12(4):641–644, 1983. doi:10.1137/0212043.
- 39 Avi Wigderson. Low-depth arithmetic circuits: technical perspective. *Communications of the ACM*, 60(6):91–92, 2017. URL: <https://cacm.acm.org/magazines/2017/6/217747-technical-perspective-low-depth-arithmetic-circuits/fulltext>.
- 40 Wikipedia. Binomial coefficient— bounds and asymptotic formulas. URL: [https://en.wikipedia.org/wiki/Binomial\\_coefficient#Bounds\\_and\\_asymptotic\\_formulas](https://en.wikipedia.org/wiki/Binomial_coefficient#Bounds_and_asymptotic_formulas).
- 41 Richard Zippel. Probabilistic algorithms for sparse polynomials. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation*, EUROSAM '79, pages 216–226, 1979. doi:10.1007/3-540-09519-5\_73.

## A Sum of powers of small support-union

We give a way to represent any univariate polynomial as sum of  $r$ -th powers of polynomials.

Here we use the notion of sumset. In additive combinatorics, the *sumset*, also called the *Minkowski sum* of two subsets  $A$  and  $B$  of an abelian group  $G$  is defined to be the set of all sums of an element from  $A$  with an element from  $B$ ,

$$A + B = \{a + b \mid a \in A, b \in B\}.$$

The  $n$ -fold iterated sumset of  $A$  is  $nA = A + \dots + A$ , where there are  $n$  summands.

We want a *small support-union* representation of a  $d$ -degree polynomial  $f$  as a sum of  $r$ -th powers, where  $r$  is constant. We consider a *small*  $B$  such that  $rB$  covers  $\{0, 1, \dots, d\}$ . Let  $t$  be the *unique* non-negative integer such that  $(t-1)^r < d+1 \leq t^r$ . Define the set  $B$  as

$$B = \{a_i t^k \mid 0 \leq a_i \leq t-1, 0 \leq k \leq r-1\}.$$

So  $|B| = rt = O(d^{1/r})$ . Let  $k \in \{0, 1, \dots, d\}$ . The base- $t$  representation of  $k$  is a sum of at most  $r$  elements from  $B$ . Hence,  $\{0, 1, \dots, d\} \subseteq rB$ .

The largest element in  $B$  is  $m := (t-1)t^{r-1}$ . Note that for any  $\epsilon > 0$ , we have  $t < (1+\epsilon)(d+1)^{1/r}$ , for all large enough  $d$ . Thus, for *any* constant  $c > 1$  and large enough  $d$ , we have  $m < c(d+1)$ . Therefore, the largest element in  $rB$  is at most  $mr < cr(d+1) = O(d)$ .

► **Lemma 19.** *Let  $\mathbb{F}$  be a field of characteristic 0 or large. For any  $f(x) \in \mathbb{F}[x]$  of degree  $d$ , there exist  $\ell_i \in \mathbb{F}[x]$  with  $\text{supp}(\ell_i) \subseteq B$  and  $c_i \in \mathbb{F}$ , for  $i = 0, 1, \dots, mr$ , such that  $f(x) = \sum_{i=0}^{mr} c_i \ell_i^r$ .*

**Proof.** Consider  $\ell_i(\mathbf{z}_i, x) = \sum_{j \in B} z_{ij} x^j$ , for distinct indeterminates  $z_{ij}$ , for all  $i, j$ . Surely,  $\deg_x(\ell_i) = m$ . There exists  $mr + 1$  many degree- $r$  polynomials  $Q_j$  over  $|B| = rt$  many variables, such that

$$\ell_i(\mathbf{z}_i, x)^r = \sum_{j=0}^{mr} Q_j(\mathbf{z}_i) x^j \quad \forall i \in [mr].$$

Note that from any monomial in  $Q_j$  we could recover  $j$  uniquely. Denote the index set  $S \subseteq [0, mr]$  such that  $Q_j \neq 0$ , for all  $j \in S$ . We could conclude that  $Q_j(\mathbf{z}_i)$  ( $j \in S$ ) are  $\mathbb{F}$ -linearly independent. We would only focus on the  $Q_j$ 's for  $j \in S$ , now onwards. Note:  $[0 \dots d] \subseteq S$ .

Suppose  $f(x) =: \sum_{i=0}^d f_i x^i$ . Define  $\tilde{f} \in \mathbb{F}^{|S|}$  and  $A \in \mathbb{F}[z]^{|S| \times |S|}$  as

$$\tilde{f} := (f_0 \quad f_1 \quad \dots \quad f_d \quad 0 \quad \dots \quad 0), \quad A := \begin{pmatrix} Q_{j_1}(\mathbf{z}_1) & Q_{j_2}(\mathbf{z}_1) & \dots & Q_{j_s}(\mathbf{z}_1) \\ Q_{j_1}(\mathbf{z}_2) & Q_{j_2}(\mathbf{z}_2) & \dots & Q_{j_s}(\mathbf{z}_2) \\ \vdots & \vdots & \dots & \vdots \\ Q_{j_1}(\mathbf{z}_{|S|}) & Q_{j_2}(\mathbf{z}_{|S|}) & \dots & Q_{j_s}(\mathbf{z}_{|S|}) \end{pmatrix}.$$

We want to find  $\mathbf{c} = (c_1 \quad c_2 \quad \dots \quad c_{|S|}) \in \mathbb{F}^{|S|}$  and  $\boldsymbol{\alpha} = (\alpha_{ij})_{i,j}$  such that

$$\sum_{j \in [|S|]} c_j \cdot \ell_j(\boldsymbol{\alpha}, x)^r = \sum_{i=0}^d f_i x^i \iff \mathbf{c} \cdot A|_{\mathbf{z}=\boldsymbol{\alpha}} \cdot \begin{pmatrix} \vdots \\ x^j \\ \vdots \end{pmatrix}_{j \in S} = \tilde{f} \cdot \begin{pmatrix} \vdots \\ x^j \\ \vdots \end{pmatrix}_{j \in S}.$$

The last expression holds  $\iff \mathbf{c} \cdot A|_{\mathbf{z}=\boldsymbol{\alpha}} = \tilde{f}$ . As the  $\mathbf{z}_i$ 's are distinct variables, the first column of  $A$  consists of different variables at each coordinate. Moreover, the first row of  $A$  contains  $\mathbb{F}$ -linearly independent  $Q_j$ 's. Thus, for *random*  $\alpha_{ij} \in \mathbb{F}$ , matrix  $A|_{\mathbf{z}=\boldsymbol{\alpha}}$  has *full rank* over  $\mathbb{F}$ . Fix such an  $\boldsymbol{\alpha}$ . This fixes  $\mathbf{c} = \tilde{f} \cdot (A|_{\mathbf{z}=\boldsymbol{\alpha}})^{-1}$ .

From the above construction, it follows that  $f(x) = \sum_{j \in [|S|]} c_j \cdot \ell_j(\boldsymbol{\alpha}, x)^r$ .  $\blacktriangleleft$

The number of *distinct* monomials across  $\ell_j(\boldsymbol{\alpha}, x)$ 's is  $|B| = O(d^{1/r})$ . While the top-fanin, as seen before, is  $\leq mr + 1 = \Theta(d)$ .

► **Remark 20.**

1. The above calculation does *not* give small support-sum representation of  $f$ , as the top-fanin is already  $\Omega(d)$ .
2. The above representation crucially requires a *field*  $\mathbb{F}$ . E.g. it does not exist for  $f_d$  over the ring  $\mathbb{Z}$ .

## B Further optimizing the top-fanin

In this section, we show a SOS- and SOC-representation for any polynomial  $f(x)$ , wherein both the top-fanin *and* the support-union size are small, namely  $O(\sqrt{d})$ . We assume that characteristic of  $\mathbb{F}$  is  $\neq 2$  in case of SOS, and  $\neq 3$ , in case of SOC. The representations are based on discussions with Agrawal [1].

### B.1 Small SOS

By Lemma 19 for  $r = 2$ , any  $f(x)$  can be written as  $f(x) = \sum_{i=1}^{O(d)} c_i f_i^2$ , with support-sum  $|\bigcup_i \text{supp}(f_i)| = O(\sqrt{d})$ . We show that the top-fanin can be reduced to  $O(\sqrt{d})$ .

► **Theorem 21** (Small SOS-Representation). *Any polynomial  $f \in \mathbb{F}[x]$  of degree  $d$  has a SOS-representation such that the top-fanin and the support-union are bounded by  $O(\sqrt{d})$ .*

The key to prove Theorem 21 is the following lemma. It shows how to decrease the top-fanin in a representation without increasing the support-union.

► **Lemma 22.** *Let  $f \in \mathbb{F}[x]$  be written as  $f = \sum_{i=1}^s c_i f_{i,1} f_{i,2}$ , with support-union  $t = |\bigcup_{i,j} \text{supp}(f_{i,j})|$ . Then there exists a representation  $f = \sum_{i=1}^t c'_i f'_{i,1} f'_{i,2}$  with support-union  $\leq t$ .*

Let us first argue why Lemma 22 implies Theorem 21. We start from the representation given by Lemma 19 mentioned above and apply Lemma 22. It follows that  $f$  can be re-written as  $f(x) = \sum_{i=1}^{O(\sqrt{d})} c'_i f_{i,1} f_{i,2}$ , where  $|\bigcup_{i,j} \text{supp}(f_{i,j})| = O(\sqrt{d})$ . This can be turned into a SOS-representation by  $f_{i,1} f_{i,2} = (f_{i,1} + f_{i,2})^2/4 - (f_{i,1} - f_{i,2})^2/4$ . Note that the last step does not change the support-union, and at most doubles the top-fanin. Thus, Theorem 21 follows.

**Proof of Lemma 22.** For the given representation of  $f$ , we assume w.o.l.g. that  $\deg(f_{i,1}) \geq \deg(f_{i,2})$  and that  $f_{i,1}, f_{i,2}$  are monic, for  $i = 1, 2, \dots, s$ . Let  $S = \bigcup_{i,j} \text{supp}(f_{i,j})$ .

We construct the representation claimed in the lemma by ensuring the following properties:

1. For every  $x^e \in S$  there is exactly one  $i$  such that  $\deg(f'_{i,1}) = e$ .
2.  $\bigcup_{i,j} \text{supp}(f'_{i,j}) \subseteq S$ .

Since we also maintain that  $\deg(f'_{i,1}) \geq \deg(f'_{i,2})$ , it follows that the top-fanin is indeed bounded by  $t = |S|$  as claimed.

We handle the monomials in  $S$  successively according to decreasing degree. Let  $x^e \in S$  be the monomial with the largest  $e$  that occurs more than once as the degree of a  $f_{i,1}$ , say  $\deg(f_{1,1}) = \deg(f_{2,1}) = e$ .

Define  $g_1 = f_{2,1} - f_{1,1}$ . Then we have  $f_{2,1} = f_{1,1} + g_1$  and  $\deg(g_1) < e$ . Moreover, the support of  $g_1$  is contained in the support of  $f_{1,1}$  and  $f_{2,1}$ . If  $\deg(f_{2,2}) = e$ , then we define similarly  $g_2 = f_{2,2} - f_{1,1}$ . Then  $f_{2,2} = f_{1,1} + g_2$  and  $\deg(g_2) < e$ . Now we can write

$$\begin{aligned} c_1 f_{1,1} f_{1,2} + c_2 f_{2,1} f_{2,2} &= c_1 f_{1,1} f_{1,2} + c_2 (f_{1,1} + g_1)(f_{1,1} + g_2) \\ &= f_{1,1} (c_1 f_{1,2} + c_2 f_{1,1} + c_2 g_1 + c_2 g_2) + c_2 g_1 g_2 \end{aligned}$$

The second line is a new sum of two products, where only the first product has terms of degree  $e$ , whereas in the second product,  $g_1, g_2$  have smaller degree. Also, the support-union set has not increased.

In case when  $\deg(f_{2,2}) < e$ , we can just work with  $f_{2,2}$  directly instead of  $f_{1,1} + g_2$ , and the above equations gets even simpler. ◀

## B.2 Small SOC

We show two small SOC-representation with different parameters. First, we show a  $\sqrt{d}$  SOC-representation that follows essentially from Theorem 21.

► **Corollary 23** ( $\sqrt{d}$  SOC-representation). *Any polynomial  $f \in \mathbb{F}[x]$  of degree  $d$  has a SOC-representation such that the top-fanin and the support-union are bounded by  $O(\sqrt{d})$ .*

**Proof.** By Theorem 21 we can write  $f$  as  $f(x) = \sum_{i=1}^{O(\sqrt{d})} c_i f_i^2$ , with support-union  $O(\sqrt{d})$ . Each  $f_i^2$  can in turn be written as  $f_i^2 = \sum_{j=1}^4 c_{i,j} (f_i + \lambda_{i,j})^3$ , for some constants  $c_{i,j}, \lambda_{i,j} \in \mathbb{F}$ , as can be shown by interpolation. This gives the representation claimed in the theorem. ◀

The second way to get a small SOC-representation uses Lemma 19 for  $r = 3$ : Any  $f(x)$  can be written as  $f(x) = \sum_{i=1}^{O(d)} c_i f_i^3$ , with support-sum  $|\bigcup_i \text{supp}(f_i)| = O(d^{1/3})$ . We show that the top-fanin can be reduced to  $O(d^{2/3})$ .

► **Theorem 24** ( $d^{2/3}$  SOC-representation). *Any polynomial  $f \in \mathbb{F}[x]$  of degree  $d$  has a SOC-representation with top-fanin  $O(d^{2/3})$  and support-union  $O(d^{1/3})$ .*

To prove Theorem 24, we show a reduction similar to Lemma 22 for sum of product-of-3.

► **Lemma 25.** *If  $f = \sum_{i=1}^s c_i f_{i,1} f_{i,2} f_{i,3}$  with support-union  $t$ , then  $f$  can be written as  $f = \sum_{i=1}^{t^2} c'_i f'_{i,1} f'_{i,2} f'_{i,3}$  with support-union  $\leq t$ .*

**Proof.** We fix the support-union set  $S$  and the monomial ordering (as seen in Lemma 22). Assume there are  $m > t^2$  many products, like  $f_{i,1} f_{i,2} f_{i,3}$ . W.l.o.g. assume  $\deg(f_{11}) = e_i$ . Rearrange  $\sum_{i \in [m]} c_i f_{i,1} f_{i,2} f_{i,3} =: f_{1,1} \cdot P + R$ , so that  $P$  is a SOS and  $R$  is a SOC without any occurrence of  $x^{e_i}$ . Apply Lemma 22, on  $P$ , to reduce its top-fanin to  $t$ . Repeat this procedure to SOC  $R$ .

Finally, the top-fanin gets upper-bounded by  $t \cdot t = t^2$ , ◀

Theorem 24 now follows by noting that any product-of-3 can be written as a sum of four cubes, by Eqn.(9); and by Lemma 19 we have  $t = O(d^{1/3})$ .

► **Lemma 26.** *For any  $f \in \mathbb{F}[x]$ , we have  $S_{\mathbb{F}}(f) \geq \min_s (U_{\mathbb{F}}(f, 4s) - 1)$ .*

**Proof Sketch.** Suppose  $f = \sum_{i=1}^s c_i f_i^2$ . Write each  $f_i^2$  as  $f_i^2 = \sum_{j=1}^4 c_{ij} (f_i + \lambda_{ij})^3$ , for distinct  $\lambda_{ij} \in \mathbb{F}$ . Thus,  $U_{\mathbb{F}}(f, 4s) \leq (\sum_{i=1}^s |f_i|_0) + 1$ . Taking minimum over  $s$  gives the desired inequality. ◀

► **Corollary 27.** *For  $s = \Omega(d^{2/3})$ , we have  $U_{\mathbb{F}}(f, s) = \Theta(d^{1/3})$ .*

## C Sum of product-of-2 decomposition

The next lemma is can be proved by standard frontier decomposition in [29].

► **Lemma 28** (Sum of product-of-2). *Let  $f(x)$  be an  $n$ -variate, homogeneous, degree  $d$  polynomial computed by a right-heavy homogeneous circuit  $\Phi$  of size  $s$ . Then, there exist polynomials  $f_{ij} \in \mathbb{F}[x]$  s.t.*

$$f(x) = \sum_{i=1}^s f_{i1} \cdot f_{i2}, \quad \text{with the following properties:} \tag{12}$$

1.  $d/3 \leq \deg(f_{i1}), \deg(f_{i2}) \leq 2d/3$ , for all  $i \in [s]$ ,
2.  $\deg(f_{i1}) + \deg(f_{i2}) = d$ , for all  $i \in [s]$ , and
3. each  $f_{ij}$  has a right-heavy homogeneous circuit of size at most  $s_2 := O(s)$ .

► **Remark 29.** For a non-homogeneous polynomial  $f(x)$ , we can apply the above for each homogeneous part of  $f(x)$ . It is well known that each homogeneous part can be computed by a homogeneous circuit of size  $O(sd^2)$ . Thus, for non-homogeneous polynomials,  $s$  can be replaced by  $O(sd^2)$  and the same conclusion follows.

## D

 Valiant's Criterion for VNP: Details for Section 3.2

A useful *sufficient* condition for a polynomial family  $(f_n(\mathbf{x}))_n$  to be in VNP is known, due to Valiant [37].

► **Theorem 30** (VNP criterion, [5]). *Let  $f_n(\mathbf{x}) = \sum_{e \in \{0,1\}^n} c_n(e) \mathbf{x}^e$  be a polynomial family such that the coefficients  $c_n(e)$  have length  $\leq n$  in binary. Then*

$$c_n(e) \in \#P/\text{poly} \implies f_n \in \text{VNP}.$$

One can further relax Theorem 30 such that the coefficients  $c_n(e)$  can actually be  $2^n$  bits long, see Theorem 13 (restated) below. The proof idea is very similar to [15, Lem. 3.2]. We also use the fact that VNP is closed under substitution. That is, for a family of polynomials  $(f(\mathbf{x}, \mathbf{y})) \in \text{VNP}$ , it also holds that  $(f(\mathbf{x}, \mathbf{y}_0)) \in \text{VNP}$ , for any value  $\mathbf{y}_0 \in \mathbb{F}^n$  assigned to the variables in  $\mathbf{y}$ .

► **Theorem 13** (restated). *Let  $f_n(\mathbf{x}) = \sum_{e \in \{0,1\}^n} c_n(e) \mathbf{x}^e$  be a polynomial family such that the coefficients  $c_n(e)$  have length  $\leq 2^n$  in binary. Let  $c_{n,j}(e)$  be the  $j$ -th bit of  $c_n(e)$ . Then*

$$c_{n,j}(e) \in \#P/\text{poly} \implies f_n \in \text{VNP}.$$

**Proof of Theorem 13.** For  $j \in \{0, 1, \dots, 2^n - 1\}$  let  $\text{bin}(j) = (j_1, \dots, j_n)$  denote the  $n$ -bit base-2 representation of  $j$  such that  $j = \sum_{i=1}^n j_i 2^{i-1}$ . Introduce new variables  $\mathbf{y} = (y_1, \dots, y_n)$  and define  $\tilde{c}_n(e, \mathbf{y}) = \sum_{j=0}^{2^n-1} c_{n,j}(e) \mathbf{y}^{\text{bin}(j)}$ . Let  $\mathbf{y}_0 := (2^{2^0}, \dots, 2^{2^{n-1}})$ . Then we have  $\tilde{c}_n(e, \mathbf{y}_0) = c_n(e)$ . Finally, consider the  $2n$ -variate auxiliary polynomial  $h_n(\mathbf{x}, \mathbf{y})$ .

$$h_n(\mathbf{x}, \mathbf{y}) = \sum_{e \in \{0,1\}^n} \tilde{c}_n(e, \mathbf{y}) \mathbf{x}^e = \sum_{e \in \{0,1\}^n} \sum_{j=0}^{2^n-1} c_{n,j}(e) \mathbf{y}^{\text{bin}(j)} \mathbf{x}^e.$$

Then we have  $h_n(\mathbf{x}, \mathbf{y}_0) = f_n(\mathbf{x})$ . Since  $c_{n,j}(e)$  can be computed in  $\#P/\text{poly}$ , we have  $(h_n(\mathbf{x}, \mathbf{y}))_n \in \text{VNP}$ . As VNP is closed under substitution, it follows that  $(f_n(\mathbf{x}))_n \in \text{VNP}$ . ◀

## E

 SOS-hardness with constant  $\varepsilon$  implies truly exponential separation between VP and VNP

We use Lemma 28 repeatedly (constant many times) to bring the degree of the intermediate polynomials “fractional”-close to  $d/2$ , namely  $d \cdot (1/2 + O(1))$ . This would be crucially used to establish the exponential separation between VP and VNP.

► **Lemma 31** (Constant boosting VSBR). *Let  $f(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$  be a degree- $d$ ,  $n$ -variate polynomial computed by homogeneous circuit of size  $s$ . Then, for any constant  $1 < \gamma < 2$ , there exist polynomials  $f_{ij} \in \mathbb{F}[\mathbf{x}]$  such that*

$$f(\mathbf{x}) = \sum_{i=1}^{\text{poly}(s)} f_{i1} \cdot f_{i2}, \text{ with the following properties} \tag{13}$$

1. each  $f_{ij}$  has a homogeneous circuit of size  $O(s)$ ,
2.  $\deg(f_{ij}) < d/\gamma$ , for all  $i, j$ ,
3.  $\deg(f_{i1}) + \deg(f_{i2}) = d$ , for all  $i$ .

**Proof sketch.** Lemma 28 shows that  $f(\mathbf{x})$  can be decomposed as  $\sum_{i=1}^s \tilde{f}_{i1} \cdot \tilde{f}_{i2}$  where  $\tilde{f}_{ij}$  has circuits of size  $O(s)$  and  $\deg(\tilde{f}_{ij}) \leq 2d/3$ , with  $\deg(\tilde{f}_{i1}) + \deg(\tilde{f}_{i2}) = d$ . Let  $\delta' := 1/\gamma - 1/2$ . Choose a constant  $m := \lceil \log_{3/2}(1/\delta') \rceil$  so that  $(2/3)^m < \delta'$ . Apply the above product-of-2 decomposition  $m$  times repeatedly on each product to conclude that  $f(\mathbf{x})$  can be decomposed as  $f(\mathbf{x}) = \sum_{i=1}^{\text{poly}(s)} g_{i1} \cdot g_{i2} \cdot \dots \cdot g_{i2^m}$ ; where  $\deg(g_{ij}) \leq (2/3)^m \cdot d < d \cdot \delta'$  and  $\text{size}(g_{ij}) = O(s)$ . Cluster each product so that the degree of each is in  $[d/2, d/\gamma]$ ; the choice of  $m$  ensures this. Hence, the conclusion follows.  $\blacktriangleleft$

Using the above fine-grained decomposition, we can prove the exponential separation between VP and VNP; the parameters change due to the different decomposition.

► **Theorem 32** (Constant  $\varepsilon$ ). *If there exists a univariate family  $(f_d(x))_d$  that is SOS-hard with some constant  $\varepsilon$ , then VNP is exponentially harder than VP ( $\mathcal{E}$  blackbox-PIT  $\in$  QP).*

## F

 Hardness to derandomization: Details for Section 3.3

Very recently, Guo et al. in [9] showed utility of the hardness of *constant* variate polynomials to derandomize PIT. To make this discussion formal, we start with the following definition.

► **Definition 33** (Hitting-set generator (HSG)). *A polynomial map  $G : \mathbb{F}^k \rightarrow \mathbb{F}^n$  given by  $G(\mathbf{z}) = (g_1(\mathbf{z}), g_2(\mathbf{z}), \dots, g_n(\mathbf{z}))$  is said to be a hitting-set generator (HSG) for a class  $\mathcal{C} \subseteq \mathbb{F}[x_1, x_2, \dots, x_n]$  of polynomials if for every nonzero  $f \in \mathcal{C}$ , we have that  $f \circ G = f(g_1, g_2, \dots, g_n)$  is nonzero.*

► **Theorem 34** ([9]). *Let  $P \in \mathbb{F}[\mathbf{x}]$  be a  $k$ -variate polynomial of degree  $d$  such that  $\text{coef}(P)$  can be computed in  $\text{poly}(d)$ -time. If  $\text{size}(P) > s^{10k+2} \cdot d^3$ , then there is a  $\text{poly}(s)$ -time HSG for  $\mathcal{C}(s, s, s)$ .*





# Circuit Depth Reductions

**Alexander Golovnev**

Georgetown University, Washington, DC, USA  
alex.golovnev@gmail.com

**Alexander S. Kulikov**

Steklov Institute of Mathematics at St. Petersburg, Russia  
St. Petersburg State University, Russia  
alexanderskulikov@gmail.com

**R. Ryan Williams**

CSAIL & EECS, MIT, Cambridge, MA, USA  
rrw@mit.edu

---

## Abstract

The best known size lower bounds against unrestricted circuits have remained around  $3n$  for several decades. Moreover, the only known technique for proving lower bounds in this model, gate elimination, is inherently limited to proving lower bounds of less than  $5n$ . In this work, we propose a non-gate-elimination approach for obtaining circuit lower bounds, via certain depth-three lower bounds. We prove that every (unbounded-depth) circuit of size  $s$  can be expressed as an OR of  $2^{s/3.9}$  16-CNFs. For DeMorgan formulas, the best known size lower bounds have been stuck at around  $n^{3-o(1)}$  for decades. Under a plausible hypothesis about probabilistic polynomials, we show that  $n^{4-\epsilon}$ -size DeMorgan formulas have  $2^{n^{1-\Omega(\epsilon)}}$ -size depth-3 circuits which are approximate sums of  $n^{1-\Omega(\epsilon)}$ -degree polynomials over  $\mathbb{F}_2$ . While these structural results do not immediately lead to new lower bounds, they do suggest new avenues of attack on these longstanding lower bound problems.

Our results complement the classical depth-3 reduction results of Valiant, which show that *logarithmic*-depth circuits of linear size can be computed by an OR of  $2^{\epsilon n} n^\delta$ -CNFs, and slightly stronger results for series-parallel circuits. It is known that no purely graph-theoretic reduction could yield interesting depth-3 circuits from circuits of super-logarithmic depth. We overcome this limitation (for small-size circuits) by taking into account both the graph-theoretic and functional properties of circuits and formulas.

We show that improvements of the following pseudorandom constructions imply super-linear circuit lower bounds for log-depth circuits via Valiant's reduction: dispersers for varieties, correlation with constant degree polynomials, matrix rigidity, and hardness for depth-3 circuits with constant bottom fan-in. On the other hand, our depth reductions show that even modest improvements of the known constructions give elementary proofs of improved (but still linear) circuit lower bounds.

**2012 ACM Subject Classification** Theory of computation → Circuit complexity

**Keywords and phrases** Circuit complexity, formula complexity, pseudorandomness, matrix rigidity

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.24

**Related Version** A full version of the paper [18] is available at <https://arxiv.org/abs/1811.04828>.

**Funding** *Alexander S. Kulikov*: Research presented in Section 4 is supported by the RNF grant 18-71-10042.

*R. Ryan Williams*: Supported by NSF CCF-1909429 and CCF-1741615.

## 1 Introduction

The Boolean circuit model is natural for computing Boolean functions. A circuit corresponds to a simple straight line program where every instruction performs a binary operation on two operands, each of which is either an input or the result of a previous instruction. The structure of this program is extremely simple: no loops, no conditional statements. Still, we



© Alexander Golovnev, Alexander S. Kulikov, and R. Ryan Williams;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 24; pp. 24:1–24:20



Leibniz International Proceedings in Informatics

LIPIcs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

know no functions in P (or even NP, or even  $E^{NP}$ ) that requires even  $3.1n$  binary instructions (“size”) to compute on inputs of length  $n$ . This is in sharp contrast with the fact that it is easy to *non-constructively* find such functions: simple counting arguments show a random function on  $n$  variables has circuit size  $\Omega(2^n/n)$  with probability  $1 - o(1)$  [52].

The strongest known circuit size lower bound  $(3 + \frac{1}{86})n - o(n)$  was proved for affine dispersers for sublinear dimension [14]. This proof, as well as all previous proofs for general circuit lower bounds against explicit functions, is based on the method of gate elimination. The main idea is to find a substitution to an input variable that eliminates sufficiently many gates from the given circuit, and then proceed by induction. While this is the most successful method known so far for proving lower bounds for unrestricted circuits, the resulting case analysis becomes increasingly tedious: when eliminating (say) 3 or 4 gates, one must consider all possible cases when two of these gates coincide. It is difficult to imagine a proof of  $5n$  lower bound using these ideas. This intuition was recently made formal in [17], where it was shown that a certain formalization of the gate elimination technique is unable to obtain a stronger than  $5n$  lower bound. Therefore we must find new approaches for proving lower bounds against circuits of unbounded depth. Let us review some of the prior results on various circuit models.

### Linear Circuits

Superlinear lower bounds are not known even for linear circuits, i.e., circuits consisting of only XOR gates (also known as  $\oplus$  gates). Note every linear function with one output has a circuit of size at most  $n - 1$ . For linear circuits, we consider *linear transformations*, multi-output functions of the form  $f(x) = Ax$  where  $A \in \mathbb{F}_2^{m \times n}$ . For a random matrix  $A \in \{0, 1\}^{n \times n}$ , the size of the smallest linear circuit computing  $Ax$  is  $\Theta(n^2/\log n)$  [33] with probability  $1 - o(1)$ , but for explicitly-constructed matrices the strongest known lower bound is  $3n - o(n)$  due to Chashkin [6]. Interestingly, Chashkin’s proof is not based on gate elimination: he first shows that the parity check matrix  $H \in \{0, 1\}^{\log n \times n}$  of the Hamming code has circuit size  $2n - o(n)$  by proving that every circuit for  $H$  has at least  $n - o(n)$  gates of out-degree at least 2.<sup>1</sup> Then he “pads”  $H$  to an  $n \times n$  matrix  $H'$  and shows that  $n - o(n)$  additional gates are needed for  $H'$ . Similarly, the best known lower bound on the complexity of linear circuits with  $\log n \leq m < o(n^2)$  outputs is  $2n + m - o(n)$  (also follows from [6]).

### Log-Depth Circuits

Nothing stronger than a  $(3 + \frac{1}{86})n - o(n)$  size lower bound is known even for circuits of depth  $O(\log n)$ . It is straightforward to show that any function that depends on all of its  $n$  variables requires depth at least  $\log n$ . One can also present an explicit function that cannot be computed by a circuit of depth smaller than  $2 \log n - o(\log n)$  using Nechiporuk’s lower bound of  $n^{2-o(1)}$  on formula size over the full binary basis [35]. Still, proving superlinear size lower bounds for circuits of depth  $O(\log n)$  remains a major open problem [56].

### Constant-Depth Circuits

Another natural and simple model of computation is bounded-depth unbounded fan-in circuits, which correspond to highly parallelizable computation. In this paper, we focus on depth-2 circuits of the form  $\text{AND} \circ \text{OR}$  (i.e., CNFs) and depth-3 circuits of the form

---

<sup>1</sup> All logarithms are base 2 unless noted otherwise.

OR  $\circ$  AND  $\circ$  OR (i.e., ORs of CNFs), where the inputs of the circuit are variables and their negations, and the gates have unbounded fan-in. Such circuits are much more structured, and therefore are easier to analyze and to prove lower bounds. For example, it is easy to show that the minimal number of clauses in a CNF computing the parity of  $n$  bits is equal to  $2^{n-1}$ , which yields an optimal lower bound for depth-2 circuits. However, already for depth 3 there is a large gap between known lower and upper bounds: it is known [10, 50] that the minimum depth-3 circuit size of a random function on  $n$  variables is  $\Theta(2^{n/2})$ , but the best known lower bound for an explicit function is  $2^{\Omega(\sqrt{n})}$  [20, 22, 39, 3, 38, 34].

Much stronger lower bounds are known for depth-3 circuits where the fan-in of the “bottom” gates (those closest to the inputs) is bounded by a parameter  $k$ . Namely, for any  $k \leq O(\sqrt{n})$ , Paturi, Saks, and Zane [39] proved a  $2^{n/k}$  lower bound for computing parity, Wolfowitz [60] proved a lower bound of  $(1 + 1/k)^{n+O(\log n)}$  for  $\text{ETHR}_{\frac{n}{k+1}}$ <sup>2</sup>, and a stronger lower bound of  $2^{\frac{\mu_k n}{k-1}}$  for  $k \geq 3$  and some constants  $\mu_k > 1$  was proven in [38] for a BCH code. For example, [38] gives a lower bound of  $2^{0.612n}$  when the bottom fan-in of the circuit is  $k = 3$ , and a lower bound of  $2^{n/10}$  for the bottom fan-in  $k = 16$ . For the case of bottom fan-in  $k = 2$ , even a  $2^{n-o(n)}$  lower bound is known [40].

A simple counting argument shows that for any constant  $k = O(1)$ , a random function requires depth-3 circuits of size  $2^{n-o(n)}$ . Calabro, Impagliazzo, and Paturi [5] construct a family of  $2^{O(n^2)}$  explicit functions, most of which require depth-3 circuits with  $k = O(1)$  of size  $2^{n-o(n)}$ . Santhanam and Srinivasan [46] improve on this by constructing such a family of functions of size  $2^{f(n)}$  for every  $f(n) = \omega(n \log n)$ .

## DeMorgan Formulas

While explicit super-linear lower bounds for *circuits* are not known, there are super-linear lower bounds for *formulas*. In this paper, we focus on the well-studied DeMorgan formulas, which are circuits where every intermediate computation is used exactly once: all gates have out-degree one, and the operations are fan-in two ANDs and ORs, with inputs being variables and their negations. The two most successful methods for proving lower bounds on DeMorgan formula size are random restrictions [54, 2, 24, 36, 21, 55] as well as Karchmer–Wigderson games and the Karchmer–Raz–Wigderson conjecture [29, 27, 26, 16, 12]. Both approaches have led to a lower bound of  $n^{3-o(1)}$  and are currently stuck at giving stronger lower bounds.

### 1.1 Valiant’s Depth Reduction

Remarkably, a classical result of Valiant from the 70’s relates three of the four models above: linear, log-depth, and constant-depth circuits. Using a depth reduction for DAGs [13], Valiant [56] shows that for any circuit of size  $cn$  and depth  $d$ , and for every integer  $k$ , one can remove at most  $\frac{2ckn}{\log d}$  wires such that the resulting circuit has depth at most  $d/2^k$ . Letting  $k$  be a sufficiently large constant, this wire-removal lemma shows how any circuit of size  $O(n)$  and depth  $O(\log n)$  can be converted into an OR  $\circ$  AND  $\circ$  OR circuit where the OR output gate has fan-in  $2^{O(n/\log \log n)}$  and the lower OR gates have fan-in  $O(n^\varepsilon)$  for any desired  $\varepsilon > 0$ . Hence, by exhibiting a function that has no depth-3 circuit with these restrictions, it follows that this function cannot be computed by circuits of linear size and logarithmic depth. Unfortunately, the best known lower bounds on depth-3 circuits (as mentioned earlier) are still too far from those required for this reduction.

<sup>2</sup>  $\text{ETHR}_{\frac{n}{k+1}}$  outputs 1 if and only if the sum of the  $n$  input bits over the integers equals  $\frac{n}{k+1}$ .

In the same paper, Valiant introduced the notion of matrix rigidity (a similar notion was independently introduced by Grigoriev [19]) and related it to the size of linear circuits of log-depth using ideas similar to those described above. Alas, the known lower bounds on matrix rigidity are also far from being able to give new lower bounds on the size of log-depth linear circuits.

## 1.2 Our Results: New Depth Reductions

The main contributions of this paper are new reductions to depth-3 circuits that work for *unrestricted* circuits and (conditionally) for super-cubic formulas, as well as new results connecting various pseudorandom objects to circuit lower bounds. In particular, we show how to express super-cubic DeMorgan formulas as subexponential-size depth-3 circuits of a certain form, under the hypothesis that DeMorgan formulas have probabilistic polynomials of non-trivial degree. This suggests an approach for improving formula size lower bounds, by proving strong lower bounds on depth-3 circuits.

### 1.2.1 Depth Reductions for Circuits

In Valiant’s depth reduction, one can only have  $d/2^k < \log n$  (and  $< cn$  removed edges) for circuits of depth  $d \leq O(\log n)$ . Thus, Valiant’s depth reduction technique does not yield interesting results for circuits of super-logarithmic depth. Moreover, Schnitger and Klawe [47, 48, 30] construct an explicit family of DAGs showing that the parameters achieved by Valiant are essentially optimal. Their counterexamples convincingly show that a pure graph-theoretic approach to circuit depth reduction cannot give non-trivial results for unrestricted circuits.

In this paper, we overcome this difficulty by presenting a counterpart of Valiant’s depth reduction that works for circuits of unrestricted depth. Our depth reduction takes into account not only the underlying graph of a circuit, but also the *functions* computed by the circuit gates.

Our first result shows that unbounded-depth circuits of size less than  $3.9n$  can be converted into  $2^{\delta n}$  disjunctions of short 16-CNFs, for some  $\delta < 1$ .

► **Theorem 1.** *Every circuit of size  $s$  can be computed as an  $OR_{2^{\lceil \frac{s}{2} \rceil}} \circ AND_s \circ OR_2$  circuit and as an  $OR_{2^{\lceil \frac{s}{3.9} \rceil}} \circ AND_{2^{14 \cdot s}} \circ OR_{16}$  circuit.*

As a consequence, in order to prove a  $3.9n - o(n)$  size lower bound on *unrestricted* circuits, it suffices to provide a function that cannot be computed by an OR of fewer than  $2^{n-o(n)}$  16-CNF’s. To prove Theorem 1, we gradually transform the given circuit into an OR of CNF’s by carefully picking a suitable internal gate and branching on its two possible output values. In contrast to Valiant’s reduction, our transformation works for circuits of arbitrary depth. This is achieved by an argument that takes into account both the graph structure of the circuit *and* the functional properties of the gates involved. Since in this approach we can branch on *internal* gates (inside the circuit), we can avoid a massive case analysis. This also distinguishes our approach from known circuit lower bound proofs based on gate elimination, which must set input gates (or gates very close to the inputs) for the argument to work.

It should be noted that known satisfiability algorithms based on branching, as well as circuit lower bounds based on gate elimination [39, 38, 49, 45, 8] may be viewed as depth-reductions for small circuits: if at most  $k$  variables are set in any branch before the circuit has a “trivial” form, then the circuit can be expressed as an OR of  $2^k$  “trivial” forms. At the same time, the known techniques in this line of work appear stuck at lower bounds of around  $3n$ , and provably cannot go beyond linear-size bounds [17].

On the way to proving Theorem 1, we study structural results about converting small circuits into disjunctions of  $k$ -CNFs, that have curious connections to properties of  $k$ -CNFs found in the Satisfiability Coding Lemma [39, 38] and Sparsification Lemma [25, 5]. In particular, we ask the following question.

► **Open Problem 2.** *Prove or disprove: for any constant  $c$ , any circuit of size  $cn$  can be computed as an*

$$OR_{2^{(1-\delta(c))n}} \circ AND \circ OR_{\gamma(c)}$$

*circuit, for some  $\delta(c) > 0$  and integer  $\gamma(c) \geq 1$ .*

If such depth-3 circuits always existed, this would constitute a new approach to proving superlinear circuit lower bounds. If no depth-3 circuit of this form exists for some linear-size circuits, then we would have a separation between linear-size circuits and (for example) super-linear-size series-parallel circuits (by Valiant’s reduction for such circuits, see Theorem 9). Note that for the gate elimination method such limitations are known [17], and they do not apply to the approach presented in this work.

Our second result is a new “non-rigidity” result for matrices with small linear circuits: if a matrix  $M$  over  $\mathbb{F}_2$  can be computed by a linear circuit of size  $s$ , then it is possible to flip at most 16 bits in every row of  $M$  to drop its rank below  $s/4$ . This opens up an approach to proving linear circuit lower bounds on sizes up to  $4n$ .

► **Theorem 3.** *For every matrix  $M \in \mathbb{F}_2^{m \times n}$  of linear circuit complexity  $s$ ,  $\mathbb{R}_M(\lfloor s/4 \rfloor) \leq 16$ .*

## 1.2.2 Pseudorandom Objects and Circuit Lower Bounds

The classical result by Valiant shows that improvements of known depth-3 circuit lower bounds and rigid matrices imply super-linear log-depth circuit lower bounds. Our depth reductions show that even modest improvements of the known constructions also give modest improvements of unrestricted circuit lower bounds.

In the full version of this paper [18], we show that Valiant’s and our reduction are applicable to two more types of pseudorandom objects: dispersers for varieties, and functions having small correlation with low degree polynomials. These implications are briefly summarized<sup>3</sup> in Table 1.

## 1.2.3 Depth Reduction for Formulas

For DeMorgan formulas we give a conditional depth-reduction (stated informally, see Theorem 14 for a formal statement): if there is an  $\varepsilon > 0$  such that DeMorgan formulas of size  $s$  have probabilistic polynomials of degree  $s^{1-\varepsilon}$  and error  $1/3$  over  $\mathbb{F}_2$ , then for some  $\delta > 0$  every DeMorgan formula of size  $O(n^{3+\delta})$  can be written as an approximate sum of  $2^{n^{1-\gamma}}$  degree- $n^{1-\gamma}$   $\mathbb{F}_2$ -polynomials for a constant  $\gamma > 0$ .<sup>4</sup> Moreover, if there are probabilistic polynomials of degree  $O(\sqrt{s})$  for DeMorgan formulas of size  $s$  (which we conjecture is true), our depth reduction holds for DeMorgan formulas of size  $n^{3.99}$ .

<sup>3</sup> In this table we only present strongest implications from the strongest premises. Our reductions would still give new circuit lower bounds even from weaker objects (see the full version [18] for complete statements of these results). For example, the second line of the table says that a lower bound of  $2^{n-o(n)}$  against depth-3 circuits would give a lower bound of  $3.9n$ . On the other hand, a lower bound of  $2^{0.8n}$  would lead to an elementary proof of a lower bound of  $3.1n$ .

<sup>4</sup> Similar results can be stated for  $\mathbb{F}_p$  where  $p$  is any prime.

■ **Table 1** Comparing the depth reductions of this paper (labeled with \*) with the depth reduction of Valiant [56] (labeled with V). We use the following notation (all formal definitions are given in Section 2 and the full version of the paper [18]):  $s(f)$  is the smallest size of a circuit computing  $f$ ,  $s_{\log}$  refers to circuits of depth  $O(\log n)$ ,  $s_3^k$  refers to circuits that are ORs of  $k$ -CNFs,  $s_{\oplus}$  refers to circuits consisting of  $\oplus$  gates only;  $(d, m, s)$ -disp. stands for a  $(d, m, s)$ -disperser, a function that is not constant on any subset of the Boolean hypercube of size at least  $s$  that is defined as the set of common roots of at most  $m$  polynomials of degree at most  $d$ ;  $\mathbb{R}_M(r)$  is the row-rigidity of  $M$  for the rank  $r$  over  $\mathbb{F}_2$ , i.e., the smallest row-sparsity of a matrix  $A$  such that  $\text{rank}(M \oplus A) \leq r$ .

	improving known lower bound	to lower bound	implies lower bound
V	$s_3^{n^\varepsilon}(f) \geq 2^{n^{1-\varepsilon}}$ [39]	$s_3^{n^\varepsilon}(f) \geq 2^{\omega\left(\frac{n}{\log \log n}\right)}$	$s_{\log}(f) = \omega(n)$
*	$s_3^{16}(f) \geq 2^{\frac{n}{10}}$ [38]	$s_3^{16}(f) \geq 2^{n-o(n)}$	$s(f) \geq 3.9n$
V	$\left(n^\varepsilon, \infty, 2^{n-n^{1/2-\varepsilon}}\right)$ -disp. [44]	$\left(n^\varepsilon, \infty, 2^{n-\omega\left(\frac{n}{\log \log n}\right)}\right)$ -disp.	$s_{\log}(f) = \omega(n)$
*	$(16, \infty, 2^{(1-\varepsilon)n})$ -disp. [58]	$(16, 1.3n, 2^{o(n)})$ -disp.	$s(f) \geq 3.9n$
*	$\left(16, \frac{n}{(\log n)^c}, 2^{o(n)}\right)$ -disp. [9]	$(16, 1.3n, 2^{o(n)})$ -disp.	$s(f) \geq 3.9n$
V	$\mathbb{R}_M\left(\omega\left(\frac{n}{\log \log n}\right)\right) > \log \log n$ [15]	$\mathbb{R}_M\left(\omega\left(\frac{n}{\log \log n}\right)\right) > n^\varepsilon$	$s_{\oplus, \log}(M) = \omega(n)$
*	$\mathbb{R}_M\left(\frac{n}{65}\right) > 16$ [41]	$\mathbb{R}_M(n - o(n)) > 16$	$s_{\oplus}(M) \geq 4n$

Interestingly, the techniques used to express DeMorgan formulas as depth-3 circuits are totally different from those used in Theorem 1 and 3. Namely, we first balance a formula (without increasing its size too much), decompose it into a small top part and several small bottom formulas, approximate the top part by a real-valued low-degree polynomial, then rewrite the bottom parts as probabilistic polynomials (as hypothesized). Finally, we collapse these two polynomials into a depth-3 circuit.

The hypothesis that lower-degree probabilistic polynomials exist for every DeMorgan formula of size  $s$  looks very plausible. We have not found an example of a size- $s$  formula that resists the construction of an  $O(\sqrt{s})$ -degree probabilistic polynomial. Note that such polynomials do exist in the real-approximation sense [43]. For example, every symmetric function (such as MAJORITY) has probabilistic polynomials of  $O(\sqrt{s})$  degree [1], and it is not hard to show that the layered OR-AND tree of depth  $\log_2(s)$  has a probabilistic polynomial of  $O(\sqrt{s})$  degree as well; in fact, *any* layered tree of depth  $\log_2(s)$  with the same gate type at each layer (AND or OR) has such degree.<sup>5</sup> It is possible that there are “nasty” formulas that resist lower-degree probabilistic polynomials, but given the examples we already know, we do not know what they might look like.

► **Open Problem 4.** *Prove or disprove: every DeMorgan formula of size  $s$  has a probabilistic polynomial over  $\mathbb{F}_2$  of degree  $O(\sqrt{s})$  with constant error less than  $1/2$ .*

<sup>5</sup> Briefly: we can always write such formulas as either an OR of ANDs of  $O(\sqrt{s})$  literals, or an AND of ORs of  $O(\sqrt{s})$  literals. From there, we can simply replace the output gate with an  $O(1)$ -degree probabilistic polynomial (as in Razborov [42]), and the other gates with exact polynomials of  $O(\sqrt{s})$  degree.



### 1.3 Motivating Example

Here we provide a simple example of a reduction of unbounded circuits to depth-3 circuits, to give an idea of what is possible.

A *formula* is a circuit where every internal gate (i.e. not the inputs and not the output) has out-degree exactly 1. In our simple example, we will show that a circuit of size, say,  $2.7n$  can be computed by an OR of  $2^{0.9n}$  formulas of small size ( $2.7n$ ). Since we know almost-quadratic lower bounds [35] on formula size, we may hope to find a function which is not computable by an OR of  $\ll 2^n$  linear-size formulas.

► **Lemma 5 (Toy Example).** *Every circuit of size  $s$  can be expressed as an OR of  $2^{\lceil s/3 \rceil}$  formulas, each of size less than  $s$ .*

**Proof.** For a circuit  $C$ , let  $s(C)$  denote its size. For  $s \leq 3$ , we just transform a circuit into a single formula of the same size. For  $s > 3$ , we proceed by induction. If the given circuit  $C$  is a formula, no transformation is needed. Otherwise take the topologically first gate  $G$  of out-degree at least 2. Note  $G$  is computed by a formula (all previous gates have out-degree 1); let  $t = s(G)$  be the size of this formula. Consider two minimum-size circuits  $C_0$  and  $C_1$  that compute the same function as  $C$  on the input sets  $\{x \in \{0,1\}^n : G(x) = 0\}$  and  $\{x \in \{0,1\}^n : G(x) = 1\}$ , respectively. We claim that  $s(C_0), s(C_1) \leq s - t - 2 \leq s - 3$ , since to compute  $C_0$  and  $C_1$  one can remove the subcircuit in  $C$  computing gate  $G$  as well as two successors of  $G$ . The successors can be removed because  $G$  outputs a constant on both parts of the considered partition of the Boolean hypercube, and all gates in the subcircuit of  $G$  are only needed to compute  $G$  ( $G$  is computed by a formula). Now, note that

$$\mathcal{C}(x) \equiv (\neg G(x) \wedge C_0(x)) \vee (G(x) \wedge C_1(x)).$$

Applying the induction hypothesis to  $C_0$  and  $C_1$ , we can rewrite  $\mathcal{C}$  as an OR of at most  $2^{\lceil (s-3)/3 \rceil} \leq 2^{\lceil s/3 \rceil}$  formulas of size  $(s - t - 2) + (t + 1) < s$ . ◀

This result would imply a circuit lower bound of  $3n - o(n)$  for any function that has correlation at most  $2^{-n+o(n)}$  with all formulas of linear size. While we do know functions that have exponentially small correlation  $2^{-\varepsilon n}$  with formulas of linear size [45, 28, 51, 31, 55, 23], none of them gives a bound of  $2^{-n+o(n)}$ . At any rate there is an inherent limitation for this toy approach. By Parseval's identity, every Boolean function has a Fourier coefficient  $\geq 2^{-n/2}$ . This implies that the correlation of this function with the corresponding parity function is at least  $2^{-n/2}$  (and this is essentially tight correlation with small formulas for a random function). Since every parity on a subset of inputs can be computed by a formula of size  $\leq n$ , Lemma 5 would only be able to prove circuit lower bounds of  $1.5n$ .

In order to prove stronger circuit lower bounds, we need to improve both parameters: the constant 3 in the exponent, and the class of formulas we reduce circuits to. Our Theorem 1 achieves this: it reduces a circuit to an OR of  $2^{\lceil \frac{s}{3.5} \rceil}$  formulas, each of which is a 16-CNF. Therefore strong enough correlation bounds against 16-CNFs would yield new circuit lower bounds.

## 2 Definitions and Preliminaries

### 2.1 Unrestricted Circuits

Let  $B_{n,m}$  be the set of all Boolean functions  $f: \{0,1\}^n \rightarrow \{0,1\}^m$  and let  $B_2 = B_{2,1}$ . A *circuit* is a directed acyclic graph that has  $n$  nodes of in-degree 0 labeled with  $x_1, \dots, x_n$  that are called *input gates*. All other nodes are called *internal gates*, have in-degree 2, and are labeled

with operations from  $B_2$ . Some  $m$  gates are also marked as output gates. Such a circuit computes a function from  $B_{n,m}$  in a natural way. The *size*  $s(\mathcal{C})$  of a circuit  $\mathcal{C}$  is its number of *internal* gates. This definition extends naturally to functions:  $s(f)$  is the smallest size of a circuit computing the function  $f$ .

The *depth* of a gate  $G$  is the maximum number of edges (also called *wires*) on a path from an input gate to  $G$ . The depth of a circuit is the maximum depth of its gates. By  $s_{\log n}(f)$  we denote the smallest size of a circuit of depth  $O(\log n)$  computing  $f$ .

A circuit is called *linear* if it consists of  $\oplus$  gates only. The corresponding circuit size measure is denoted by  $s_{\oplus}$ .

Our unrestricted circuits are usually drawn with input gates at the top, so by a top gate of a circuit we mean a gate that is fed by two variables.

## 2.2 Series-Parallel Circuits

A *labeling* of a directed acyclic graph  $G = (V, E)$  is a function  $\ell: V \rightarrow \mathbb{N}$  such that for every edge  $(u, v) \in E$  one has  $\ell(u) < \ell(v)$ . A graph/circuit  $G$  is called *series-parallel* if there exists a labeling  $\ell$  such that for no two edges  $(u, v), (u', v') \in E$ ,  $\ell(u) < \ell(u') < \ell(v) < \ell(v')$ . The corresponding circuit complexity measure is  $s_{\text{sp}}$ .

## 2.3 Depth-3 Circuits

Unlike unrestricted circuits, depth-3 circuits are usually drawn the other way around, i.e., with the output gate at the top. In this paper, we focus on  $\text{OR} \circ \text{AND} \circ \text{OR}$  circuits, i.e., ORs of CNFs. We will use subscripts to indicate the fact that the fan-in of a particular layer is bounded. Namely, an  $\text{OR}_p \circ \text{AND}_q \circ \text{OR}_r$  circuit is an OR of at most  $p$  CNFs each of which contains at most  $q$  clauses and at most  $r$  literals in every clause. Since the gates of a depth 3 circuit are allowed to have an unbounded fan-in, it is natural to define the size of such a circuit as its number of wires. It is not difficult to see that for  $k = O(1)$  the size of an  $\text{OR} \circ \text{AND} \circ \text{OR}_k$  circuit is equal to the fan-in of its output gate up to a polynomial factor in  $n$ . By  $s_3^k(f)$  we denote the smallest size of an  $\text{OR} \circ \text{AND} \circ \text{OR}_k$  circuit computing  $f$ .

## 2.4 Rigidity

We say that a matrix  $M \in \mathbb{F}_2^{m \times n}$  is *s-sparse* if each *row* of  $M$  contains at most  $s$  non-zero elements. The *rigidity* of a matrix  $M \in \mathbb{F}_2^{m \times n}$  for the rank parameter  $r$  is the minimum sparsity of a matrix  $A \in \{0, 1\}^{m \times n}$  such that  $\text{rank}_{\mathbb{F}_2}(M \oplus A) \leq r$ :

$$\mathbb{R}_M(r) = \min\{s: \text{rank}_{\mathbb{F}_2}(M \oplus A) \leq r, A \text{ is } s\text{-sparse}\}.$$

## 2.5 Probabilistic, Approximate, and Robust Polynomials

Since even functions of small circuit and formula complexity may only have large-degree polynomial representations, it often proves convenient to use randomized polynomials or polynomials which approximate (rather than exactly compute) a given function.

► **Definition 6** (Probabilistic polynomials). *Let  $f: \{0, 1\}^n \rightarrow \{0, 1\}$  be a Boolean function. A distribution  $\mathcal{D}$  of  $n$ -variate degree- $d$  polynomials over  $\mathbb{F}_2$  is a probabilistic polynomial for  $f$  with degree  $d$  and error  $\varepsilon$  if for every  $x \in \{0, 1\}^n$ ,*

$$\Pr_{p \sim \mathcal{D}}[f(x) = p(x)] \geq 1 - \varepsilon.$$

► **Definition 7** (Approximate Polynomials). Let  $f: \{0, 1\}^n \rightarrow \{0, 1\}$  be a Boolean function. An  $n$ -variate multilinear degree- $d$  polynomial  $p$  over  $\mathbb{R}$  is an approximate polynomial for  $f$  with degree  $d$  and error  $\varepsilon$  if for every  $x \in \{0, 1\}^n$ ,

$$|p(x) - f(x)| \leq \varepsilon.$$

► **Definition 8** (Robust Polynomials). Let  $f: \{0, 1\}^n \rightarrow [0, 1]$  be a polynomial over  $\mathbb{R}$ . Then a polynomial  $p: \mathbb{R}^n \rightarrow \mathbb{R}$  is  $\delta$ -robust for  $f$  if for every  $x \in \{0, 1\}^n$  and for every  $\varepsilon \in [-1/3, 1/3]^n$ ,

$$|f(x) - p(x + \varepsilon)| \leq \delta.$$

## 2.6 Valiant's Depth Reductions

Here we formally recall the classical depth reduction results by Valiant [56].

► **Theorem 9** ([56, 4, 57]). For every  $c \geq 1$  and  $\varepsilon > 0$  there exists a  $\delta > 0$  such that every circuit  $\mathcal{C}$  of size  $cn$  and depth  $c \log n$  can be computed as

1. an  $OR_{\frac{\delta n}{2^{\log \log n}}} \circ AND \circ OR_{n^\varepsilon}$  circuit
2. and as an  $OR_{2^{\varepsilon n}} \circ AND \circ OR_{2^{(\log n)^{1-\delta}}}$  circuit.

Furthermore, for every  $c \geq 1$  and  $\varepsilon > 0$  there is a  $k \geq 1$  such that every series-parallel circuit of size  $cn$  and unbounded depth can be computed as an  $OR_{2^{\varepsilon n}} \circ AND \circ OR_k$  circuit.

Theorem 9 applied to linear circuits yields the following.

► **Theorem 10** ([56, 4, 57]). Let  $M \in \mathbb{F}^{m \times n}$  be a matrix. For every  $c \geq 1$  and  $\varepsilon > 0$  there exists  $\delta > 0$  such that, if a linear circuit  $\mathcal{C}$  of size  $cn$  and depth  $c \log n$  computes  $Mx$  for every  $x \in \mathbb{F}^n$ , then

1.  $\mathbb{R}_M \left( \frac{\delta n}{\log \log n} \right) \leq n^\varepsilon$ ;
2. and  $\mathbb{R}_M(\varepsilon n) \leq 2^{(\log n)^{1-\delta}}$ .

Furthermore, for every  $c \geq 1$  and  $\varepsilon > 0$  there is a  $k \geq 1$  such that if  $\mathcal{C}$  is a series-parallel linear circuit of size  $cn$  and unbounded depth, then  $\mathbb{R}_M(\varepsilon n) \leq k$ .

## 3 Formula Depth Reduction

In this section, we give a (conditional) depth reduction for DeMorgan formulas. We start by balancing a given formula. For this we use the following result due to Tal [55].

► **Lemma 11** (Claim VI.2 in [55]). Let  $F$  be a DeMorgan formula of size  $s$  over the set of variables  $X = \{x_1, \dots, x_n\}$ , and  $t$  be some parameter; then, there exist  $k \leq 36s/t$  formulas over  $X$ , denoted by  $T_1, \dots, T_k$ , each of size at most  $t$ , and there exists a read-once formula  $F'$  of size  $k$  such that  $F'(T_1(x), \dots, T_k(x)) = F(x)$  for all  $x \in \{0, 1\}^n$ .

Below we will also make use of the following results by Reichardt [43] and Sherstov [53].

► **Theorem 12** ([43]). If  $f: \{0, 1\}^n \rightarrow \{0, 1\}$  can be computed by a DeMorgan formula of size  $s$ , then  $f$  has an approximate polynomial of degree  $O(\sqrt{s})$  with error  $\varepsilon = 1/10$ .

► **Theorem 13** ([53]). If  $f: \{0, 1\}^n \rightarrow [0, 1]$  is a polynomial of degree  $d$  over  $\mathbb{R}$ , then there is a  $\delta$ -robust polynomial  $p$  for  $f$  of degree  $O(d + \log(1/\delta))$ .

## 24:10 Circuit Depth Reductions

Now we are ready to present the main result of this section: Assuming DeMorgan formulas of size  $s$  have probabilistic polynomials of degree  $O(s^{1-\delta})$  for some  $\delta > 0$ , we will obtain subexponential-size depth-3 circuits computing formulas of super-cubic size.

In the following, a **SUM** gate will compute an *approximate sum*: a (real-weighted) sum of the inputs such that, over all Boolean inputs, the sum is within  $\pm 1/3$  of the 0-1 value of a desired Boolean function.

► **Theorem 14.** *Suppose for some  $\delta > 0$ , DeMorgan formulas of size  $\ell$  have probabilistic polynomials of degree  $\ell^{1-\delta}$  with error  $1/3$ . Then for every  $\alpha < \delta/(1-\delta)$  there is a  $\gamma > 0$ , so that for every formula  $F$  of size  $s = O(n^{3+\alpha})$ , there is a  $2^{n^{1-\gamma}}$ -size approximate sum of degree- $n^{1-\gamma}$   $\mathbb{F}_2$ -polynomials computing  $F$ . That is,  $F$  can be computed by a*

$$\text{SUM}_{2^{n^{1-\gamma}}} \circ \text{MOD2}_{2^{n^{1-\gamma}}} \circ \text{AND}_{n^{1-\gamma}}.$$

**Proof.** First, we apply Lemma 11 to  $F$  for some parameter  $t$  to be defined later. We obtain a read-once formula  $F'$  of size  $k = O(s/t)$ , and  $k$  formulas  $T_1, \dots, T_k$  each of size  $\leq t$ .

Let  $p$  be an approximate polynomial (over the reals) for  $F'$  of degree  $d = O(\sqrt{k})$  with error  $1/10$ , guaranteed by Theorem 12. Applying Theorem 13, we get a  $1/10$ -robust polynomial  $p'$  for  $p$  of degree  $d' = O(\sqrt{k})$ .

By the hypothesis of the theorem, we know that each  $T_i$  has a probabilistic polynomial of degree  $O(t^{1-\delta})$  with error  $\varepsilon = 1/3$ . For each  $T_i$ , draw  $O(\log s)$  independent copies of this probabilistic polynomial, and take their majority vote with an  $O(\log s)$ -degree polynomial. For an appropriate leading constant in the big-O, we can obtain a probabilistic polynomial for  $T_i$  of degree  $O(t^{1-\delta} \cdot \log s)$  with error  $1/(10s)$ .

Let  $\mathcal{D}_1, \dots, \mathcal{D}_k$  be probabilistic polynomials of degree  $D = O(t^{1-\delta} \cdot \log s)$  with error  $\varepsilon = 1/(10s)$  for the formulas  $T_1, \dots, T_k$ . The error bound  $\varepsilon = 1/(10s)$  guarantees that for every  $x \in \{0, 1\}^n$ , all  $k$  polynomials compute the correct value with probability at least  $9/10$ .

Now for every  $T_i$ , we compute the average  $A_i$  (over the reals) of  $O(n)$  independent samples from  $\mathcal{D}_i$ . By a Chernoff bound and union bound, each  $A_i$  is within  $\pm 1/10$  of the correct 0-1 value for  $T_i$ , over all  $2^n$  inputs  $x$ , with probability of error  $1/\exp(n)$ . By the properties of robust polynomials,  $p'$  fed the sums  $A_i$  will still output the correct value (within  $\pm 1/10$ ) for all inputs  $x \in \{0, 1\}^n$ , for some choice of samples.

Therefore  $F$  can be computed by a

$$\text{SUM}_{n^{d'}} \circ \text{PRODUCT}_{d'} \circ \text{SUM}_{O(n)} \circ \text{MOD2} \circ \text{AND}_D.$$

Applying distributivity to the PRODUCT of SUMs, we get

$$\text{SUM}_{n^{d'}} \circ \text{SUM}_{n^{O(d')}} \circ \text{PRODUCT}_{d'} \circ \text{MOD2} \circ \text{AND}_D.$$

Noting the PRODUCTs now take 0/1 inputs, we can replace them with ANDs:

$$\text{SUM}_{n^{d'}} \circ \text{SUM}_{n^{O(d')}} \circ \text{AND}_{d'} \circ \text{MOD2} \circ \text{AND}_D.$$

Taking the Fourier expansion of the AND function, we can replace each AND gate with a SUM of  $2^{d'}$  MOD2s of fan-in  $\leq d'$ :

$$\text{SUM}_{n^{d'}} \circ \text{SUM}_{n^{O(d')}} \circ \text{SUM}_{2^{d'}} \circ \text{MOD2} \circ \text{AND}_D.$$

Merging the SUMs, our final expression has the form:

$$\text{SUM}_{n^{O(d')}} \circ \text{MOD2} \circ \text{AND}_D.$$

Finally, we want to choose a value of  $t$  so that the fan-in of the SUM is subexponential, and the fan-ins of the AND's are sublinear (which will also imply that the fan-in of the MOD2's are sub-exponential). Let  $t = n^{1+\beta}$ , where  $\beta$  is an arbitrary number between  $\alpha < \beta < \delta/(1-\delta)$ . Note that

$$d' = O(\sqrt{k}) = O(\sqrt{s/t}) = O(n^{1-\frac{\beta-\alpha}{2}}) = O(n^{1-\gamma})$$

for every  $0 < \gamma < \frac{\beta-\alpha}{2}$ . Also, observe that

$$D = O(t^{1-\delta} \cdot \log s) = O(n^{1-(1-\delta)(\delta/(1-\delta)-\beta)} \log n) = O(n^{1-\gamma})$$

for every  $0 < \gamma < (1-\delta)(\delta/(1-\delta)-\beta)$ .

From the upper bounds on  $d'$  and  $D$ , we have that  $F$  can be computed by

$$\text{SUM}_{2^{n^{1-\gamma}}} \circ \text{MOD2}_{2^{n^{1-\gamma}}} \circ \text{AND}_{n^{1-\gamma}}$$

for some  $\gamma > 0$ . ◀

The above formula depth reduction shows that, if there are more efficient probabilistic polynomials for DeMorgan formulas (and we have no reason to doubt this), then super-cubic formulas have interesting representations as approximate sums of sub-exponentially many sub-linear degree  $\mathbb{F}_2$ -polynomials. Recent work [59, 7] can already be applied to prove interesting lower bounds against approximate sums of  $2^{n^\alpha}$   $\mathbb{F}_2$ -polynomials of degree  $n^\beta$ , where  $\alpha + \beta < 1$ . The remaining challenge will be to prove lower bounds when  $\max\{\alpha, \beta\} < 1$ .

## 4 Circuit Depth Reductions

In this section, we present new depth reductions for circuits with unrestricted depth.

### 4.1 Linear Circuits

We start by considering *linear* circuits, i.e., circuits consisting of  $\oplus$  gates only. For technical reasons, we assume that there are  $n+1$  input gates in a linear circuit:  $x_1, \dots, x_n$  as well as the constant 0. For a matrix  $M \in \{0,1\}^{m \times n}$ , we say that a linear circuit  $\mathcal{C}$  with  $m$  outputs computes the linear transformation  $M$  if the  $i$ -th output of  $\mathcal{C}(x)$  equals the  $i$ -th row of  $Mx$  for all  $x \in \{0,1\}^n$ , treating  $\mathcal{C}(x)$  as the vector of output values. We say that a linear circuit  $\mathcal{C}$  computing  $M$  is *optimal* if no circuit of smaller size computes  $M$ .

The main result of this subsection asserts that matrices computable by small linear circuits are not too rigid. The contrapositive says: to get an improved lower bound on the size of linear circuits, it suffices to construct a matrix with good rigidity parameters. Below, we restate the corresponding theorem formally and then prove it.

► **Theorem 3.** *For every matrix  $M \in \mathbb{F}_2^{m \times n}$  of linear circuit complexity  $s$ ,  $\mathbb{R}_M(\lfloor s/4 \rfloor) \leq 16$ .*

**Proof.** Let  $\mathcal{C}$  be an optimal circuit of size  $s$  computing  $M$ . If  $s < 16$  or the depth of  $\mathcal{C}$  is at most 4, then each output depends on at most 16 variables. Hence  $M$  is 16-sparse and the theorem statement holds. Consider this as the base case of an induction on  $s$ .

For the induction step, we “normalize”  $\mathcal{C}$ . Namely, we show how to express  $M$  as the (modulo 2) sum of two  $\mathbb{F}_2$ -matrices  $A$  and  $B$ , where  $A$  is 16-sparse (each row has  $\leq 16$  ones) and  $B$  has rank at most  $\lfloor s/4 \rfloor$ . Note that if  $\mathcal{C}$  has an output gate  $H$  of depth at most 4, then  $H$  depends on at most  $2^4 = 16$  inputs. Thus the corresponding row  $r_H$  of  $M$  has at most 16 ones. Consider the  $(m-1) \times n$  matrix  $M_{-H}$  obtained by removing  $r_H$  from  $M$ . We claim

that  $\mathbb{R}_{M-H}(\lfloor s/4 \rfloor) \leq 16$  implies  $\mathbb{R}_M(\lfloor s/4 \rfloor) \leq 16$ . Indeed, suppose  $M_{-H} = A_{-H} \oplus B_{-H}$  where  $A_{-H}$  is 16-sparse and  $\text{rank}(B_{-H}) \leq \lfloor s/4 \rfloor$ . To get matrices  $A$  and  $B$  for  $M$ , we simply add the row  $r_H$  to  $A_{-H}$  and a corresponding all-zero row to  $B_{-H}$ . Clearly, the resulting matrix  $A$  is 16-sparse and the rank of the resulting matrix  $B$  does not change. Thus, in the following, we assume WLOG that  $\mathcal{C}$  has no output gates of depth at most 4. Our crucial step is the following claim.

▷ **Claim 15.** Let  $\mathcal{C}$  be an optimal linear circuit computing  $M \in \{0,1\}^{m \times n}$  such that  $s(\mathcal{C}) \geq 16$ , and no output gate of  $\mathcal{C}$  has depth smaller than 5. Then there is a gate  $G$  in  $\mathcal{C}$  and a linear circuit  $\mathcal{C}'$  computing a matrix  $M' \in \{0,1\}^{m \times n}$  with the properties:

1.  $s(\mathcal{C}') \leq s(\mathcal{C}) - 4$ , and
2. for every  $x \in \{0,1\}^n$ , if  $G(x) = 0$  then  $\mathcal{C}(x) = \mathcal{C}'(x)$ .

For now, suppose the claim is proved. Consider the circuit  $\mathcal{C}'$ , gate  $G$  in  $\mathcal{C}$ , and matrix  $M'$  provided by Claim 15. Let  $g \in \{0,1\}^{1 \times n}$  be the characteristic vector of the linear function computed by  $G$ , so that  $G(x) = gx$ . By the claim,  $gx = 0$  implies  $(M \oplus M')x = 0$ . Hence  $(M \oplus M')$  is either the zero matrix, or it defines the same linear subspace as  $g$ :  $M \oplus M' = tg$  for a vector  $t \in \{0,1\}^{m \times 1}$ .

By the induction hypothesis,  $M' = A' \oplus B'$  where  $A'$  is 16-sparse, and  $\text{rank}(B') \leq \lfloor \frac{s-4}{4} \rfloor = \lfloor \frac{s}{4} \rfloor - 1$ . Thus,  $M = A' \oplus B$ , where the matrix  $B = B' \oplus tg$  has rank at most  $\lfloor s/4 \rfloor$  by subadditivity of the rank function. ◀

We now turn to proving the remaining claim.

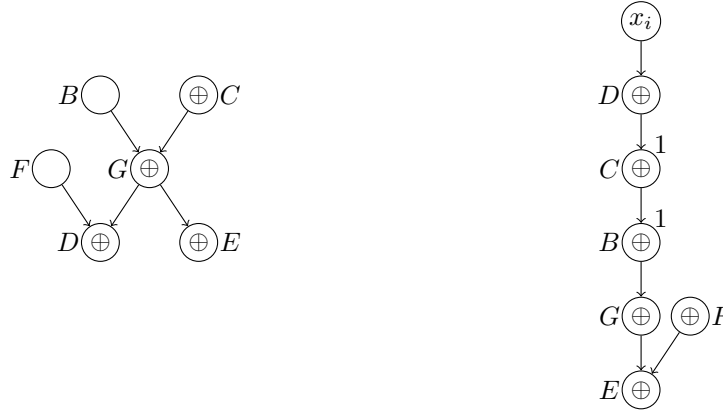
Proof of Claim 15.

**Case 1: There is a gate  $G$  in  $\mathcal{C}$  of depth at least 2 and at most 4, and has out-degree at least 2.** Let the predecessors of  $G$  be  $B$  and  $C$ , and call two of its successors  $D$  and  $E$ , see Figure 1 (in this and the following figures, we write the out-degrees of some of the gates near them). The circuit  $\mathcal{C}'$  is obtained from  $\mathcal{C}$  by “assigning” the output of  $G$  to be 0. Note that  $B(x) = C(x)$  for all  $x \in \{0,1\}^n$  where  $G(x) = 0$ . At least one of  $B$  and  $C$  must be an internal gate (otherwise  $G$  would have depth 1), let it be  $C$ . Since  $C$  computes the same function as  $B$ , it may be removed from  $\mathcal{C}'$ : we remove it, and replace every wire of the form  $C \rightarrow H$  by a new wire  $B \rightarrow H$ . Note that neither  $G$  nor  $C$  is an output gate. Now, we show that both  $D$  and  $E$  can also be removed. Let us focus on the gate  $D$  (for  $E$  it is shown similarly) and call its other predecessor  $F$ . Since  $G = 0$ , the gate  $D$  computes the same function as  $F$ . This means that one may remove  $D$ : we remove it and replace every wire  $D \rightarrow H$  by a wire  $F \rightarrow H$ . If  $D$  happens to be an output gate, we move the corresponding output label from  $D$  to  $F$ .

**Case 2: All gates of depth at least 2 and at most 4 have out-degree exactly 1 in  $\mathcal{C}$ .** Take a gate  $G$  of depth 4 and trace back its longest path to an input:  $x_i \rightarrow D \rightarrow C \rightarrow B \rightarrow G$ . Let also  $E$  be the successor of  $G$  (which exists because  $\mathcal{C}$  has depth at least 5). By assumption, gates  $B$  and  $C$  have out-degree 1. This means that in  $\mathcal{C}$  they are only used for computing the gate  $G$ . This, in turn, means that assuming  $G = 0$ , we can remove  $G$ ,  $B$ , and  $C$  (note none of them is an output). Finally, the gate  $E$  can be replaced by the other input  $F$  of  $E$  (note  $F \notin \{B, C, G\}$ , since  $\mathcal{C}$  is optimal).

This completes the proof. ◀

► **Remark 16.** Extending the same ideas, one can show that any linear circuit  $\mathcal{C}$  of size  $s$  can be computed by an  $\text{OR}_{2^{\lceil \frac{s}{4} \rceil}} \circ \text{AND}_{s \cdot 2^{14}} \circ \text{OR}_{16}$  circuit. For this, one considers two optimal circuits  $\mathcal{C}_0$  and  $\mathcal{C}_1$  resulting from  $\mathcal{C}$  by assuming  $G = 0$  and  $G = 1$ , respectively. As shown in the proof, both  $\mathcal{C}_0$  and  $\mathcal{C}_1$  have size at most  $s - 4$ . One then proceeds by induction. We illustrate this approach in full detail in the next subsection.



Case 1: assuming  $G = 0$ , the gate  $G$  is removed,  $B$  is replaced by  $C$ , and  $D$  and  $E$  are replaced by their other predecessors.

Case 2: assuming  $G = 0$ , the gates  $B$ ,  $C$ , and  $G$  are removed whereas  $E$  is replaced by  $F$ .

■ **Figure 1** Cases in the proof of Claim 15.

► **Remark 17.** The proof of Theorem 3 gives a decomposition  $M = A \oplus B = A \oplus (C \cdot D)$ , where  $A \in \mathbb{F}^{m \times n}$  is 16-sparse,  $C \in \mathbb{F}^{m \times s/4}$  is composed of vectors  $t$ , and  $D \in \mathbb{F}^{s/4 \times n}$  is composed of vectors  $g$ . Since the chosen gate  $G$  always has depth at most four, the vector  $g$  is 16-sparse. Thus, we in fact have a decomposition  $M = A \oplus (C \cdot D)$ , where both  $A$  and  $D$  are 16-sparse. In particular, the row-space of  $M$  is spanned by the union of row-spaces of  $A$  and  $D$ . This implies that the row-space of  $M$  can be spanned by at most  $(m + \frac{s}{4})$  16-sparse vectors. The corresponding matrix property is called *outer dimension*, and it is studied in [37, 32]. While the current lower bounds on the outer dimension of explicit matrices do not lead to new circuit lower bounds, it would be interesting to study their applications in this context.

## 4.2 General Boolean Circuits

In this section, we study the following natural question: given a Boolean circuit<sup>6</sup> and given an integer  $k \geq 2$ , what is the smallest  $\text{OR} \circ \text{AND} \circ \text{OR}_k$  circuit computing the same function? To this end, we introduce the following notation. For an integer  $k \geq 2$ , we define  $\alpha(k)$  as the infimum of all values  $\alpha$  such that any circuit of size  $s$  can be rewritten as a  $\text{OR}_{2^{\alpha s}} \circ \text{AND} \circ \text{OR}_k$  circuit.

For proving upper bounds on  $\alpha(k)$  it will be convenient to consider the following class of circuits. Let  $\text{OR}_p \circ \text{AND}_q \circ C(r)$  be a class of circuits with an output OR that is fed by at most  $p$  AND's of at most  $q$  circuits of size at most  $r$ .

► **Theorem 18.** *Every circuit of size  $s$  can be computed as:*

1. an  $\text{OR}_{2^{\lceil \frac{s}{2} \rceil}} \circ \text{AND}_{\lceil \frac{s}{2} \rceil} \circ C(1)$  circuit;
2. an  $\text{OR}_{2^{\lceil \frac{s}{3.9} \rceil}} \circ \text{AND}_{\lceil \frac{s}{3} \rceil} \circ C(15)$  circuit.

<sup>6</sup> In this section we consider functions with one output, but these results can be trivially generalized to the multi-output case.



## 24:14 Circuit Depth Reductions

Note that any circuit of size  $r$  depends on at most  $r + 1$  variables, and hence can be written as an  $(r + 1)$ -CNF with at most  $2^r$  clauses. Therefore every  $\text{OR}_p \circ \text{AND}_q \circ C(r)$  circuit can be easily converted into a  $\text{OR}_p \circ \text{AND}_{q2^r} \circ \text{OR}_{r+1}$  circuit. Theorem 1, which we restate below, is then an immediate corollary of Theorem 18. In turn, it implies that  $\alpha(2) \leq \frac{1}{2}$  and  $\alpha(16) \leq \frac{1}{3.9}$ .

► **Theorem 1.** *Every circuit of size  $s$  can be computed as an  $\text{OR}_{2^{\lceil \frac{s}{2} \rceil}} \circ \text{AND}_s \circ \text{OR}_2$  circuit and as an  $\text{OR}_{2^{\lceil \frac{s}{3.9} \rceil}} \circ \text{AND}_{2^{14} \cdot s} \circ \text{OR}_{16}$  circuit.*

**Proof of Theorem 18.** Both parts are proven in a similar fashion. We proceed by induction on  $s$ . The base case is when  $s$  is small. We then just have an  $\text{OR}_1 \circ \text{AND}_1 \circ C(s)$  circuit.

For the induction step we take a gate  $G$  of  $\mathcal{C}$  and consider two circuits  $\mathcal{C}_0$  and  $\mathcal{C}_1$  where  $\mathcal{C}_i$  computes the same as  $\mathcal{C}$  on all inputs  $\{x \in \{0, 1\}^n : G(x) = i\}$ . We may assume both  $\mathcal{C}_i$ 's are minimal size among all such circuits. Since  $\mathcal{C}_i$  can be obtained from  $\mathcal{C}$  by removing the gate  $G$  (as it computes the constant  $i$  on the corresponding subset of the Boolean hypercube), we conclude that  $s(\mathcal{C}_i) < s$ . This allows us to proceed by induction. Assume that by the induction hypothesis  $\mathcal{C}_i$  is guaranteed to be expressible as an  $\text{OR}_{p_i} \circ \text{AND}_{q_i} \circ C(r_i)$  circuit. We use the following identity to convert  $\mathcal{C}$  into the required circuit:

$$\mathcal{C}(x) \equiv ([G(x) = 0] \wedge \mathcal{C}_0(x)) \vee ([G(x) = 1] \wedge \mathcal{C}_1(x)). \quad (1)$$

Assume that the subcircuit of  $\mathcal{C}$  computing the gate  $G$  has at most  $t$  gates. We claim that  $[G(x) = i] \wedge \mathcal{C}_i$  can be written as an  $\text{OR}_{p_i} \circ \text{AND}_{q_i+1} \circ C(\max\{r_i, t\})$  circuit. For this, we just feed a new circuit computing  $G$  to every AND gate. Plugging this into (1), gives an

$$\text{OR}_{p_0+p_1} \circ \text{AND}_{\max\{q_0, q_1\}+1} \circ C(\max\{t, r_0, r_1\}) \quad (2)$$

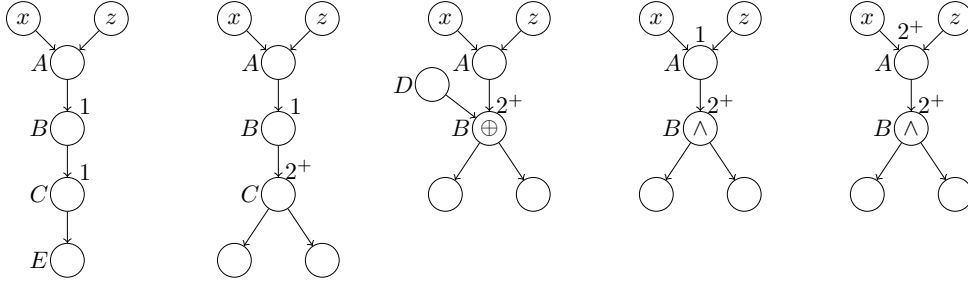
circuit for computing  $\mathcal{C}$ .

Below, we provide details specific to each of the two items from the theorem statement. In particular, we estimate the parameters  $p_i$ 's,  $q_i$ 's,  $r_i$ 's, and  $t$  and plug them into (2).

1. The base case is  $s = 1$ . Then  $\mathcal{C}$  consists of a single gate and can be expressed as an  $\text{OR}_1 \circ \text{AND}_1 \circ C(1)$  circuit. For the induction step, assume that  $s \geq 2$  and take a gate  $A$  that depends on two variables. Let  $G = A$ , hence  $t = 1$ . The gate  $A$  must have at least one successor (otherwise  $\mathcal{C}$  can be replaced by a circuit with smaller than  $s$  gates). Clearly,  $A$  and its successors are not needed in  $\mathcal{C}_i$ 's. Hence, by the induction hypothesis  $p_i \leq 2^{\frac{s-2}{2}+1}$ ,  $q_i \leq \frac{s-2}{2} + 1$ ,  $r_i \leq 1$ . Plugging this into (2) gives the desired result.
2. Take a gate  $A$  that is fed by two variables  $x$  and  $z$  and has the maximum distance to an output. If its distance to output is at most 4, then  $s(\mathcal{C}) \leq 15$  and we just rewrite it as an  $\text{OR}_1 \circ \text{AND}_1 \circ C(15)$  circuit. This is the base case. Assume now that the distance from  $A$  to the output gate is at least 5. In the analysis below, we always “follow” the longest path from  $A$  to the output. This allows us to conclude that any such path is long enough and hence each gate considered has positive out-degree (i.e., is not an output). Moreover, each gate on this path cannot depend on too many variables. Let  $B$  be a successor of  $A$  on the longest path to the output.

In the five cases below, we show that we can always find a gate  $G$  that  $s(G) \leq 15$  and both  $s(\mathcal{C}_0)$  and  $s(\mathcal{C}_1)$  are small enough. In particular,  $s(\mathcal{C}_0), s(\mathcal{C}_1) \leq s - 4$  works for us:  $p_0 + p_1 \leq 2 \cdot 2^{\lceil \frac{s-4}{3.9} \rceil} < 2^{\lceil \frac{s}{3.9} \rceil}$ ,  $\max\{q_0, q_1\} + 1 \leq \lceil \frac{s-4}{3} \rceil + 1 < \lceil \frac{s}{3} \rceil$ .

See Figure 2 for an illustration of the five cases. For a gate  $G$ , by  $\text{out}(G)$  we denote the out-degree of  $G$ .



Case 1.1: when  $E$  is constant, one removes  $B$ ,  $C$ ,  $E$ , and successors of  $E$ . Case 1.2: when  $C$  is constant, one removes  $B$ ,  $C$ , and successors of  $C$ . Case 2.1: when  $B$  is constant, one removes  $B$  and its successors, replace  $A$  by  $D \oplus c$ . Case 2.2.1: when  $B$  is constant, one removes  $B$  and its successors, and  $A$ . Case 2.2.2: when  $B$  is constant, one removes  $B$  and its successors; moreover,  $B = 1$  it forces  $A$  to be a constant and removes  $A$  and its successors.

■ **Figure 2** Cases in the proof of the second part of Theorem 18.

**Case 1:**  $\text{out}(B) = 1$ . Let  $C$  be the successor of  $B$ .

**Case 1.1:**  $\text{out}(C) = 1$ . Let  $E$  be the successor of  $C$ . Let  $G = E$ . In  $\mathcal{C}_i$ 's, one removes  $B$ ,  $C$  (as they were only needed to compute  $E$  that is now a constant),  $E$ , and the successors of  $E$ .

**Case 1.2:**  $\text{out}(C) \geq 2$ . Let  $G = C$ . In  $\mathcal{C}_i$ 's, one removes  $B$ ,  $C$ , and the successors of  $C$ .

**Case 2:**  $\text{out}(B) \geq 2$ . Let  $D$  be the other input of  $B$ . It may be a gate or an input variable. If  $B$  computes a constant Boolean binary operation or an operation that depends on  $A$  or  $D$  only, then  $C$  is not optimal. Otherwise,  $B$  computes one of the following two types of functions (either linear or quadratic polynomial over  $\mathbb{F}_2$ ):

**Case 2.1:**  $B(A, D) = A \oplus D \oplus c$  where  $c \in \{0, 1\}$ . Let  $G = B$ . In  $\mathcal{C}_i$ 's, one immediately removes  $B$  and its successors. Also, in  $\mathcal{C}_i$ ,  $D \oplus A = i \oplus c$ . Hence,  $A$  may be replaced by  $D \oplus i \oplus c$ .

**Case 2.2:**  $B(A, D) = (A \oplus a) \cdot (D \oplus d) \oplus c$  where  $a, d, c \in \{0, 1\}$ .

**Case 2.2.1:**  $\text{out}(A) = 1$ . Let  $G = C$ . In  $\mathcal{C}_i$ 's, one removes  $B$ , its successors, and  $A$ .

**Case 2.2.2:**  $\text{out}(A) \geq 2$ . Let  $D$  be the other successor of  $B$ . Let  $G = B$ . In  $\mathcal{C}_i$ 's, one removes  $B$  and its successors. Also,  $B = c \oplus 1$  forces  $A = a \oplus 1$  and  $D = d \oplus 1$ . Hence, in  $\mathcal{C}_{c \oplus 1}$  two additional gates are removed:  $A$  and its successors (if a successor of  $B$  happens to be a successor of  $A$  also, then it is a function on  $A$  and  $D$  and the circuit can be simplified, which contradicts its optimality). Hence,  $p_0 + p_1 \leq 2^{\lceil \frac{s-3}{3.9} \rceil} + 2^{\lceil \frac{s-5}{3.9} \rceil}$ . This is smaller than  $2^{\lceil \frac{s}{3.9} \rceil}$  since  $2^{-\frac{3}{3.9}} + 2^{-\frac{5}{3.9}} < 1$ .

This completes the proof. ◀

► **Remark 19.** It is not difficult to see that the output OR gate is a “disjoint OR”, and can be replaced by a SUM gate over the integers. In other words, for every  $x \in \{0, 1\}^n$ , at most one subcircuit feeding into the OR gate may evaluate to 1. This holds because we always consider two mutually exclusive cases:  $G = 0$  or  $G = 1$ .

### 4.3 Properties of $\alpha(k)$

We start by observing a lower bound on  $\alpha(k)$ .

► **Lemma 20.** *For any integer  $k \geq 2$ ,  $\alpha(k) \geq 1/k$ .*

**Proof.** Let  $\oplus_n$  denote the parity function of  $n$  inputs. It has  $2^{n-1}$  inputs where it is equal to 1 and all these inputs are isolated, that is, the Hamming distance between any pair of them is at least 2. As proven by Paturi, Pudlák, and Zane [39], every  $k$ -CNF has at most  $2^{n(1-1/k)}$  isolated satisfying assignments. This implies that  $\oplus_n$  cannot be computed by an OR of fewer than  $2^{n/k-1}$   $k$ -CNFs. Since  $s(\oplus_n) = n - 1$ , this implies that

$$\alpha(k) \geq \frac{\frac{n}{k} - 1}{n - 1}.$$

Since this must hold for arbitrary large  $n$ ,  $\alpha(k) \geq 1/k$ . ◀

Thus, we know the exact value of  $\alpha(2) = \frac{1}{2}$ . This immediately implies a circuit lower bound of  $2n - o(n)$  for BCH codes. Indeed, it was shown in [40] that when the bottom fan-in is restricted to  $k = 2$ , then BCH codes require depth-3 circuits of size  $2^{n-o(n)}$ . And, since  $\alpha(2) = \frac{1}{2}$ , they must have circuit complexity at least  $2n - o(n)$ .

One can use techniques from Theorem 18 to prove an upper bound of  $\alpha(3) \leq \frac{\log_2 3}{4}$ . Thus, we know that

$$\frac{1}{3} \leq \alpha(3) \leq \frac{\log_2 3}{4} < 0.3963.$$

We conjecture that the upper bound on  $\alpha_3$  is tight. One way to prove this would be to find the  $s_3^3$  complexity of the inner product function:  $\text{IP}(x_1, \dots, x_n) = x_1x_2 \oplus x_3x_4 \oplus \dots \oplus x_{n-1}x_n$ . In particular, if the upper bound shown in the next lemma is tight, then  $\alpha(3) = \frac{\log_2 3}{4}$ .

► **Lemma 21.**

1.  $2^{\frac{n}{4}} \leq s_3^2(\text{IP}) \leq 2^{\frac{n}{2}-o(n)}$ .
2.  $2^{\frac{n}{6}} \leq s_3^3(\text{IP}) \leq 3^{\frac{n}{4}}$ .

**Proof.** Note that by substituting every other input of IP by 1, one gets the parity function  $\oplus_{\frac{n}{2}}$  on the remaining  $n/2$  inputs. Now both lower bounds follow from the corresponding lower bounds for the parity function:  $s_3^2(\oplus_k) \geq 2^{\frac{k}{2}}$  and  $s_3^3(\oplus_k) \geq 2^{\frac{k}{3}}$ .

1. The first upper bound follows from the fact that  $\text{IP}(x_1, \dots, x_n) = 1$  iff there is an odd number of ones among

$$p_1 = x_1x_2, p_2 = x_3x_4, \dots, p_{\frac{n}{2}} = x_{n-1}x_n.$$

Hence,

$$\text{IP}(x_1, \dots, x_n) \equiv \bigvee_{S \subseteq [\frac{n}{2}]: |S| \bmod 2 = 1} \left( \bigwedge_{i \in S} [p_i = 1] \wedge \bigwedge_{i \notin S} [p_i = 0] \right).$$

It remains to note that each  $[p_i = c]$  can be expressed as a 2-CNF because  $p_i$  depends on two variables.

2. For the second upper bound, note that  $\text{IP}(x_1, \dots, x_n) = 1$  iff there is an odd number of 1's among

$$p_1 = x_1x_2 \oplus x_3x_4, p_2 = x_5x_6 \oplus x_7x_8, \dots, p_{\frac{n}{4}} = x_{n-3}x_{n-2} \oplus x_{n-1}x_n.$$

To compute IP by a depth 3 circuit, we go through all possible  $2^{\frac{n}{4}-1}$  values of  $p_1, \dots, p_{\frac{n}{4}}$  such that an odd number of them is equal to 1:

$$\text{IP}(x_1, \dots, x_n) \equiv \bigvee_{S \subseteq [\frac{n}{4}]: |S| \bmod 2 = 1} \left( \bigwedge_{i \in S} [p_i = 1] \wedge \bigwedge_{i \notin S} [p_i = 0] \right) \quad (3)$$

Now, we show that  $[p_i = 0]$  can be written as a single 3-CNF, whereas  $[p_i = 1]$  can be expressed as an OR of two 3-CNFs. W.l.o.g. assume that  $i = 1$ . The clauses of a 3-CNF expressing  $[p_i = 0]$  should reject all assignments to  $x_1, x_2, x_3, x_4 \in \{0, 1\}$  where  $\text{IP}(x_1, x_2, x_3, x_4) = 1$ . In all such assignments, one of the two monomials ( $x_1 x_2$  and  $x_3 x_4$ ) is equal to 0 whereas the other one is equal to 1. Hence, one needs to write down a set of clauses rejecting the following four partial assignments:  $\{x_1 = 0, x_3 = x_4 = 1\}$ ,  $\{x_2 = 0, x_3 = x_4 = 1\}$ ,  $\{x_1 = x_2 = 1, x_3 = 0\}$ ,  $\{x_1 = x_2 = 1, x_4 = 0\}$ . Thus,

$$[p_1(x_1, x_2, x_3, x_4) = 0] \equiv (x_1 \vee \neg x_3 \vee \neg x_4) \wedge (x_2 \vee \neg x_3 \vee \neg x_4) \wedge (\neg x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_4).$$

In turn, to express  $[p_1 = 1]$  as an OR of two 3-CNFs we consider both assignments to  $x_1$ :

$$[p_1(x_1, x_2, x_3, x_4) = 1] \equiv ((x_1) \wedge [x_2 \oplus x_3 x_4 = 0]) \vee ((\neg x_1) \wedge [x_3 x_4 = 1]).$$

It remains to note that each of  $[x_2 \oplus x_3 x_4 = 0]$  and  $[x_3 x_4 = 1]$  can be written as a 3-CNF. Let  $[p_i = 0] \equiv P_i$  and  $[p_i = 1] \equiv ((x_i) \wedge Q_i) \vee ((\neg x_i) \wedge R_i)$  where  $P_i$ ,  $Q_i$ , and  $R_i$  are 3-CNFs. One may then expand (3) as follows:

$$\bigvee_{S \subseteq [\frac{n}{4}]: |S| \bmod 2 = 1} \left( \bigvee_{T \subseteq S} \left( \bigwedge_{i \in T} ((x_i) \wedge Q_i) \wedge \bigwedge_{i \in S \setminus T} ((\neg x_i) \wedge R_i) \wedge \bigwedge_{i \notin S} P_i \right) \right)$$

The fan-in of the resulting OR-gate is

$$\sum_{S \subseteq [\frac{n}{4}]: |S| \bmod 2 = 1} 2^{|S|} \leq \sum_{i=0}^{\frac{n}{4}} \binom{n/4}{i} 2^i = 3^{\frac{n}{4}}. \quad \blacktriangleleft$$

► **Open Problem 22.** Determine  $s_3^3(\text{IP})$ .

Besides finding the exact values of  $\alpha(k)$ , it would be interesting to find out whether every circuit of *linear size* can be computed by a non-trivial depth 3 circuit with constant bottom fan-in. We restate this open problem below.

► **Open Problem 2.** Prove or disprove: for any constant  $c$ , any circuit of size  $cn$  can be computed as an

$$\text{OR}_{2^{(1-\delta(c))n}} \circ \text{AND} \circ \text{OR}_{\gamma(c)}$$

circuit, for some  $\delta(c) > 0$  and integer  $\gamma(c) \geq 1$ .

This paper supports the conjecture by showing that it holds for small values of  $c$ . As another example, we can consider a class of functions where we know linear *upper bounds* on circuit complexity. For any *symmetric* function  $f$  (i.e., a function whose value depends only on the sum over integers of the input bits) we know that  $s(f) \leq 4.5n + o(n)$  [11]. It is also known [40, 60] that symmetric functions can be computed by relatively small depth-3 circuits:  $s_3^k(f) \leq \text{poly}(n) \cdot (1 + 1/k)^n$  (and this bound is tight [60]).

Since in our depth reduction results, we always get  $k$ -CNFs with small linear number of clauses, it is interesting to study the expressiveness of OR of exponential number of such  $k$ -CNFs. Let us define  $\alpha(k, c)$  as the infimum of all values  $\alpha$  such that any circuit of size at most  $cn$  can be computed as an  $\text{OR}_{2^{\alpha n}} \circ \text{AND}_{cn} \circ \text{OR}_k$ . We can upper bound the rate of convergence of  $\alpha(k, c)$  using the following width reduction result for CNF-formulas [49, 5].

► **Theorem 23** ([49, 5]). *For any constant  $0 < \varepsilon \leq 1$  and a function  $C: \mathbb{N} \rightarrow \mathbb{N}$ , any CNF formula  $f$  with  $n$  variables and  $n \cdot C(n)$  clauses can be expressed as  $f = \text{OR}_{i=1}^t f_i$ , where  $t \leq 2^{\varepsilon n}$  and each  $f_i$  is a  $k$ -CNF formula with at most  $n \cdot C(n)$  clauses, where  $k = O\left(\frac{1}{\varepsilon} \cdot \log\left(\frac{C(n)}{\varepsilon}\right)\right)$ .*

For our applications, we are interested in  $\alpha(k, c)$  for small fixed  $c$ . Since for every  $c$ ,  $\alpha(k, c)$  is a non-increasing bounded sequence, we let  $\alpha(\infty, c) = \lim_{k \rightarrow \infty} \alpha(k, c)$ . Then Theorem 23 implies that  $\alpha(k, c) \geq \alpha(\infty, c) \geq \alpha(k, c) - O\left(\frac{\log(c/k)}{k}\right)$ .

---

## References

- 1 Josh Alman and Ryan Williams. Probabilistic polynomials and hamming nearest neighbors. In *FOCS 2015*, pages 136–150. IEEE, 2015.
- 2 Alexander E. Andreev. On a method for obtaining more than quadratic effective lower bounds for the complexity of  $\pi$ -schemes. *Moscow Univ. Math. Bull.*, 42(1):63–66, 1987.
- 3 Ravi B. Boppana. The average sensitivity of bounded-depth circuits. *Inf. Process. Lett.*, 63(5):257–261, 1997.
- 4 Chris Calabro. A lower bound on the size of series-parallel graphs dense in long paths. In *ECCC*, volume 15, 2008.
- 5 Chris Calabro, Russell Impagliazzo, and Ramamohan Paturi. A duality between clause width and clause density for SAT. In *CCC 2006*, pages 252–260, 2006.
- 6 Aleksandr V. Chashkin. On the complexity of Boolean matrices, graphs and their corresponding Boolean functions. *Discrete Math. and Appl.*, 4(3):229–257, 1994.
- 7 Lijie Chen and Ryan Williams. Circuit lower bounds from PCP of proximity. *Unpublished manuscript*, 2019.
- 8 Ruiwen Chen and Valentine Kabanets. Correlation bounds and #SAT algorithms for small linear-size circuits. In *COCOON 2015*, pages 211–222. Springer, 2015.
- 9 Gil Cohen and Avishay Tal. Two structural results for low degree polynomials and applications. In *RANDOM 2015*, pages 680–709, 2015.
- 10 Vlado Dančík. Complexity of Boolean functions over bases with unbounded fan-in gates. *Inf. Process. Lett.*, 57(1):31–34, 1996.
- 11 Evgeny Demenkov, Arist Kojevnikov, Alexander S. Kulikov, and Grigory Yaroslavtsev. New upper bounds on the boolean circuit complexity of symmetric functions. *Inf. Process. Lett.*, 110(7):264–267, 2010.
- 12 Irit Dinur and Or Meir. Toward the KRW Composition Conjecture: Cubic Formula Lower Bounds via Communication Complexity. In *CCC 2016*, pages 3:1–3:51, 2016.
- 13 Paul Erdős, Ronald L. Graham, and Endre Szemerédi. On sparse graphs with dense long paths. *Comp. and Math. with Appl.*, 1:145–161, 1975.
- 14 Magnus G. Find, Alexander Golovnev, Edward A. Hirsch, and Alexander S. Kulikov. A better-than- $3n$  lower bound for the circuit complexity of an explicit function. In *FOCS 2016*, pages 89–98, 2016.
- 15 Joel Friedman. A note on matrix rigidity. *Combinatorica*, 13(2):235–239, 1993.
- 16 Dmitry Gavinsky, Or Meir, Omri Weinstein, and Avi Wigderson. Toward better formula lower bounds: An information complexity approach to the KRW composition conjecture. In *STOC 2014*, pages 213–222, 2014.
- 17 Alexander Golovnev, Edward A. Hirsch, Alexander Knop, and Alexander S. Kulikov. On the limits of gate elimination. *J. Comput. Syst. Sci.*, 96:107–119, 2018.

- 18 Alexander Golovnev, Alexander S. Kulikov, and R. Ryan Williams. Circuit depth reductions. *arXiv*, 2018. [arXiv:1811.04828](#).
- 19 Dmitrii Yu. Grigoriev. Application of separability and independence notions for proving lower bounds of circuit complexity. *Zap. Nauch. Sem. POMI*, 60:38–48, 1976.
- 20 Johan Håstad. Almost optimal lower bounds for small depth circuits. In *STOC 1986*, pages 6–20, 1986.
- 21 Johan Håstad. The shrinkage exponent of de Morgan formulas is 2. *SIAM J. Comput.*, 27(1):48–64, 1998.
- 22 Johan Håstad, Stasys Jukna, and Pavel Pudlák. Top-down lower bounds for depth 3 circuits. In *FOCS 1993*, pages 124–129, 1993.
- 23 Russell Impagliazzo and Valentine Kabanets. Fourier concentration from shrinkage. *Comput. Complex.*, 26(1):275–321, 2017.
- 24 Russell Impagliazzo and Noam Nisan. The effect of random restrictions on formula size. *Random Struct. Algorithms*, 4(2):121–134, 1993.
- 25 Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001.
- 26 Mauricio Karchmer, Ran Raz, and Avi Wigderson. Super-logarithmic depth lower bounds via the direct sum in communication complexity. *Comput. Complex.*, 5(3/4):191–204, 1995.
- 27 Mauricio Karchmer and Avi Wigderson. Monotone circuits for connectivity require super-logarithmic depth. *SIAM J. Discrete Math.*, 3(2):255–265, 1990.
- 28 Tali Kaufman, Shachar Lovett, and Ely Porat. Weight distribution and list-decoding size of reed–muller codes. *IEEE Trans. Inf. Theory*, 58(5):2689–2696, 2012.
- 29 Valeriy M. Khrapchenko. A method of determining lower bounds for the complexity of  $\pi$ -schemes. *Math. Notes of the Acad. of Sci. of the USSR*, 10(1):474–479, 1971.
- 30 Maria M. Klawe. Shallow grates. *Theor. Comput. Sci.*, 123(2):389–395, 1994.
- 31 Ilan Komargodski, Ran Raz, and Avishay Tal. Improved average-case lower bounds for demorgan formula size. In *FOCS 2013*, pages 588–597, 2013.
- 32 Satyanarayana V. Lokam. Complexity lower bounds using linear algebra. *Found. Trends Theor. Comput. Sci.*, 4(1-2):1–155, 2009.
- 33 Oleg B. Lupanov. On rectifier and switching-and-rectifier schemes. *Dokl. Akad. Nauk SSSR*, 111(6):1171–1174, 1956. In Russian.
- 34 Or Meir and Avi Wigderson. Prediction from partial information and hindsight, with application to circuit lower bounds. In *ECCC*, volume 24, 2017.
- 35 Edward I. Nechiporuk. On a Boolean function. *Dokl. Akad. Nauk SSSR*, 169(4):765–766, 1966.
- 36 Mike Paterson and Uri Zwick. Shrinkage of de Morgan formulae under restriction. *Random Struct. Algorithms*, 4(2):135–150, 1993.
- 37 Ramamohan Paturi and Pavel Pudlák. Circuit lower bounds and linear codes. *J. Math. Sci.*, 134(5):2425–2434, 2006.
- 38 Ramamohan Paturi, Pavel Pudlák, Michael E Saks, and Francis Zane. An improved exponential-time algorithm for  $k$ -SAT. *J. ACM*, 52(3):337–364, 2005.
- 39 Ramamohan Paturi, Pavel Pudlák, and Francis Zane. Satisfiability coding lemma. In *FOCS 1997*, pages 566–574, 1997.
- 40 Ramamohan Paturi, Michael E. Saks, and Francis Zane. Exponential lower bounds for depth 3 Boolean circuits. In *STOC 1997*, pages 86–91, 1997.
- 41 Pavel Pudlák and Zdeněk Vavřín. Computation of rigidity of order  $\frac{n^2}{r}$  for one simple matrix. *Comment. Math. Univ. Carolinae*, 32(2):213–218, 1991.
- 42 Alexander A. Razborov. Lower bounds on the dimension of schemes of bounded depth in a complete basis containing the logical addition function. *Mat. Zametki*, 41(4):598–607, 1987.
- 43 Ben W. Reichardt. Reflections for quantum query algorithms. In *SODA 2011*, pages 560–569. SIAM, 2011.
- 44 Zachary Remscrem. The Hilbert function, algebraic extractors, and recursive fourier sampling. In *FOCS 2016*, pages 197–208, 2016.

- 45 Rahul Santhanam. Fighting perebor: New and improved algorithms for formula and QBF satisfiability. In *FOCS 2010*, pages 183–192, 2010.
- 46 Rahul Santhanam and Srikanth Srinivasan. On the limits of sparsification. In *ICALP 2012*, pages 774–785, 2012.
- 47 Georg Schnitger. A family of graphs with expensive depth-reduction. *Theor. Comput. Sci.*, 18(1):89–93, 1982.
- 48 Georg Schnitger. On depth-reduction and grates. In *FOCS 1983*, pages 323–328, 1983.
- 49 Rainer Schuler. An algorithm for the satisfiability problem of formulas in conjunctive normal form. *J. Algorithms*, 54(1):40–44, 2005.
- 50 Igor S. Sergeev. On complexity of circuits and formulas of bounded depth over unbounded fan-in bases. *Disc. Math. Appl.*, 30(2):120–137, 2018. In Russian.
- 51 Kazuhisa Seto and Suguru Tamaki. A satisfiability algorithm and average-case hardness for formulas over the full binary basis. *Comput. Complex.*, 22(2):245–274, 2013.
- 52 Claude E. Shannon. The synthesis of two-terminal switching circuits. *Bell Syst. Tech. J.*, 28:59–98, 1949.
- 53 Alexander A Sherstov. Making polynomials robust to noise. In *STOC 2012*, pages 747–758. ACM, 2012.
- 54 Bella A. Subbotovskaya. Realizations of linear functions by formulas using  $+$ ,  $\cdot$ ,  $-$ . *Dokl. Akad. Nauk SSSR*, 136(3):553–555, 1961.
- 55 Avishay Tal. Shrinkage of De Morgan formulae by spectral techniques. In *FOCS 2014*, pages 551–560. IEEE, 2014.
- 56 Leslie G. Valiant. Graph-theoretic arguments in low-level complexity. In *MFCS 1977*, pages 162–176, 1977.
- 57 Emanuele Viola. On the power of small-depth computation. *Found. Trends Theor. Comput. Sci.*, 5(1):1–72, 2009.
- 58 Emanuele Viola and Avi Wigderson. Norms, XOR lemmas, and lower bounds for polynomials and protocols. *Theory Comput.*, 4(1):137–168, 2008.
- 59 Richard Ryan Williams. Limits on representing Boolean functions by linear combinations of simple functions: Thresholds, ReLUs, and low-degree polynomials. In *CCC 2018*, pages 6:1–6:24, 2018.
- 60 Guy Wolfowitz. The complexity of depth-3 circuits computing symmetric Boolean functions. *Inf. Process. Lett.*, 100(2):41–46, 2006.



# Dynamic Inference in Probabilistic Graphical Models

**Weiming Feng**

State Key Laboratory for Novel Software Technology, Nanjing University, China  
fengwm@smail.nju.edu.cn

**Kun He**

Shenzhen University, China  
Shenzhen Institute of Computing Sciences, China  
hekun.threebody@foxmail.com

**Xiaoming Sun** 

State Key Laboratory of Computer Architecture, Institute of Computing Technology,  
Chinese Academy of Sciences, Beijing, China  
University of Chinese Academy of Sciences, Beijing, China  
sunxiaoming@ict.ac.cn

**Yitong Yin**

State Key Laboratory for Novel Software Technology, Nanjing University, China  
yinyt@nju.edu.cn

---

## Abstract

Probabilistic graphical models, such as Markov random fields (MRFs), are useful for describing high-dimensional distributions in terms of local dependence structures. The probabilistic inference is a fundamental problem related to graphical models, and sampling is a main approach for the problem. In this paper, we study probabilistic inference problems when the graphical model itself is changing dynamically with time. Such dynamic inference problems arise naturally in today's application, e.g. multivariate time-series data analysis and practical learning procedures.

We give a dynamic algorithm for sampling-based probabilistic inferences in MRFs, where each dynamic update can change the underlying graph and all parameters of the MRF simultaneously, as long as the total amount of changes is bounded. More precisely, suppose that the MRF has  $n$  variables and polylogarithmic-bounded maximum degree, and  $N(n)$  independent samples are sufficient for the inference for a polynomial function  $N(\cdot)$ . Our algorithm dynamically maintains an answer to the inference problem using  $\tilde{O}(nN(n))$  space cost, and  $\tilde{O}(N(n) + n)$  incremental time cost upon each update to the MRF, as long as the Dobrushin-Shlosman condition is satisfied by the MRFs. This well-known condition has long been used for guaranteeing the efficiency of Markov chain Monte Carlo (MCMC) sampling in the traditional static setting. Compared to the static case, which requires  $\Omega(nN(n))$  time cost for redrawing all  $N(n)$  samples whenever the MRF changes, our dynamic algorithm gives a  $\tilde{\Omega}(\min\{n, N(n)\})$ -factor speedup. Our approach relies on a novel dynamic sampling technique, which transforms local Markov chains (a.k.a. single-site dynamics) to dynamic sampling algorithms, and an “algorithmic Lipschitz” condition that we establish for sampling from graphical models, namely, when the MRF changes by a small difference, samples can be modified to reflect the new distribution, with cost proportional to the difference on MRF.

**2012 ACM Subject Classification** Theory of computation → Design and analysis of algorithms; Computing methodologies → Machine learning

**Keywords and phrases** Dynamic inference, probabilistic graphical model, Gibbs sampling, Markov random field

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.25

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/1904.11807>.



© Weiming Feng, Kun He, Xiaoming Sun, and Yitong Yin;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 25; pp. 25:1–25:20



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

**Funding** Weiming Feng and Yitong Yin are supported by the National Key R&D Program of China 2018YFB1003202 and the National Natural Science Foundation of China under Grant Nos. 61722207 and 61672275. Kun He and Xiaoming Sun are supported by the National Natural Science Foundation of China Grants No. 61832003, 61433014 and K.C.Wong Education Foundation.

## 1 Introduction

The probabilistic graphical models provide a rich language for describing high-dimensional distributions in terms of the dependence structures between random variables. The *Markov random field* (MRF) is a basic graphical model that encodes pairwise interactions of complex systems. Given a graph  $G = (V, E)$ , each vertex  $v \in V$  is associated with a function  $\phi_v : Q \rightarrow \mathbb{R}$ , called the *vertex potential*, on a finite domain  $Q = [q]$  of  $q$  *spin states*, and each edge  $e \in E$  is associated with a symmetric function  $\phi_e : Q^2 \rightarrow \mathbb{R}$ , called the *edge potential*, which describes a pairwise interaction. Together, these induce a probability distribution  $\mu$  over all configurations  $\sigma \in Q^V$ :

$$\mu(\sigma) \propto \exp(H(\sigma)) = \exp\left(\sum_{v \in V} \phi_v(\sigma_v) + \sum_{e=\{u,v\} \in E} \phi_e(\sigma_u, \sigma_v)\right).$$

This distribution  $\mu$  is known as the Gibbs distribution and  $H(\sigma)$  is the *Hamiltonian*. It arises naturally from various physical models, statistics or learning problems, and combinatorial problems in computer science [29, 25].

The *probabilistic inference* is one of the most fundamental computational problems in graphical model. Some basic inference problems ask to calculate the marginal distribution, conditional distribution, or maximum-a-posteriori probabilities of one or several random variables [37]. Sampling is perhaps the most widely used approach for probabilistic inference. Given a graphical model, independent samples are drawn from the Gibbs distribution and certain statistics are computed using the samples to give estimates for the inferred quantity. For most typical inference problems, such statistics are easy to compute once the samples are given, for instance, for estimating the marginal distribution on a variable subset  $S$ , the statistics is the frequency of each configuration in  $Q^S$  among the samples, thus the cost for inference is dominated by the cost for generating random samples [24, 35].

The classic probabilistic inference assumes a static setting, where the input graphical model is fixed. In today's application, dynamically changing graphical models naturally arise in many scenarios. In various practical algorithms for learning graphical models, e.g. the contrastive divergence algorithm for learning the restricted Boltzmann machine [22] and the iterative proportional fitting algorithm for maximum likelihood estimation of graphical models [37], the optimal model  $\mathcal{I}^*$  is obtained by updating the parameters of the graphical model iteratively (usually by gradient descent), which generates a sequence of graphical models  $\mathcal{I}_1, \mathcal{I}_2, \dots, \mathcal{I}_M$ , with the goal that  $\mathcal{I}_M$  is a good approximation of  $\mathcal{I}^*$ . Also in the study of the multivariate time-series data, the dynamic Gaussian graphical models [5], multiregression dynamic model [32], dynamic graphical model [14], and dynamic chain graph models [2], are all dynamically changing graphical models and have been used in a variety of applications. Meanwhile, with the advent of Big Data, scalable machine learning systems need to deal with continuously evolving graphical models (see e.g. [33] and [34]).

The theoretical studies of probabilistic inference in dynamically changing graphical models are lacking. In the aforementioned scenarios in practice, it is common that a sequence of graphical models is presented with time, where any two consecutive graphical models can differ from each other in all potentials but by a small total amount. Recomputing the

inference problem from scratch at every time when the graphical model is changed, can give the correct solution, but is very wasteful. A fundamental question is whether probabilistic inference can be solved dynamically and efficiently.

In this paper, we study the problem of probabilistic inference in an MRF when the MRF itself is changing dynamically with time. At each time, the whole graphical model, including all vertices and edges as well as their potentials, are subject to changes. Such *non-local* updates are very general and cover all applications mentioned above. The problem of *dynamic inference* then asks to maintain a correct answer to the inference in a dynamically changing MRF with low incremental cost proportional to the amount of changes made to the graphical model at each time.

## 1.1 Our results

We give a dynamic algorithm for sampling-based probabilistic inferences. Given an MRF instance with  $n$  vertices, suppose that  $N(n)$  independent samples are sufficient to give an approximate solution to the inference problem, where  $N : \mathbb{N}^+ \rightarrow \mathbb{N}^+$  is a polynomial function. We give dynamic algorithms for general inference problems on dynamically changing MRF.

Suppose that the current MRF has  $n$  vertices and polylogarithmic-bounded maximum degree, and each update to the MRF may change the underlying graph and/or all vertex/edge potentials, as long as the total amount of changes is bounded. Our algorithm maintains an approximate solution to the inference with  $\tilde{O}(nN(n))$  space cost, and with  $\tilde{O}(N(n) + n)$  incremental time cost upon each update, assuming that the MRFs satisfy the Dobrushin-Shlosman condition [8, 9, 7]. The condition has been widely used to imply the efficiency of Markov chain Monte Carlo (MCMC) sampling (e.g. see [19, 12]). Compared to the static algorithm, which requires  $\Omega(nN(n))$  time for redrawing all  $N(n)$  samples each time, our dynamic algorithm significantly improves the time cost with an  $\tilde{\Omega}(\min\{n, N(n)\})$ -factor speedup.

On specific models, the Dobrushin-Shlosman condition has been established in the literature, which directly gives us following efficient dynamic inference algorithms, with  $\tilde{O}(nN(n))$  space cost and  $\tilde{O}(N(n) + n)$  time cost per update, on graphs with  $n$  vertices and maximum degree  $\Delta = O(1)$ :

- for Ising model with temperature  $\beta$  satisfying  $e^{-2|\beta|} > 1 - \frac{2}{\Delta+1}$ , which is close to the uniqueness threshold  $e^{-2|\beta_c|} = 1 - \frac{2}{\Delta}$ , beyond which the static versions of sampling or marginal inference problem for anti-ferromagnetic Ising model is intractable [17, 16];
- for hardcore model with fugacity  $\lambda < \frac{2}{\Delta-2}$ , which matches the best bound known for sampling algorithm with near-linear running time on general graphs with bounded maximum degree [36, 28, 13];
- for proper  $q$ -coloring with  $q > 2\Delta$ , which matches the best bound known for sampling algorithm with near-linear running time on general graphs with bounded maximum degree [23].

Our dynamic inference algorithm is based on a dynamic sampling algorithm, which efficiently maintains  $N(n)$  independent samples for the current MRF while the MRF is subject to changes. More specifically, we give a dynamic version of the *Gibbs sampling* algorithm, a local Markov chain for sampling from the Gibbs distribution that has been studied extensively. Our techniques are based on: (1) couplings for dynamic instances of graphical models; and (2) dynamic data structures for representing single-site Markov chains so that the couplings can be realized algorithmically in sub-linear time. Both these techniques are of independent interest, and can be naturally extended to more general settings with multi-body interactions.

Our results show that on dynamically changing graphical models, sampling-based probabilistic inferences can be solved significantly faster than rerunning the static algorithm at each time. This has practical significance in speeding up the iterative procedures for learning graphical models.

## 1.2 Related work

The problem of dynamic sampling from graphical models was introduced very recently in [14]. There, a dynamic sampling algorithm was given for graphical models with soft constraints, and can only deal with local updates that change a single vertex or edge at each time. The regimes for such dynamic sampling algorithm to be efficient are much more restrictive than the conditions for the rapid mixing of Markov chains. Our algorithm greatly improves the regimes for efficient dynamic sampling for the Ising and hardcore models in [14], and for the first time, can handle non-local updates that change all vertex/edge potentials simultaneously. Besides, the dynamic/online sampling from log-concave distributions was also studied in [31, 26].

Another related topic is the dynamic graph problems, which ask to maintain a solution (e.g. spanners [15, 30, 38] or shortest paths [3, 21, 20]) while the input graph is dynamically changing. More recently, important progress has been made on dynamically maintaining structures that are related to graph random walks, such as spectral sparsifier [11, 1] or effective resistances [10, 18]. Instead of one particular solution, dynamic inference problems ask to maintain an estimate of a statistics, such statistics comes from an exponential-sized probability space described by a dynamically changing graphical model.

## 1.3 Organization of the paper

In Section 2, we formally introduce the dynamic inference problem. In Section 3, we formally state the main results. Preliminaries are given in Section 4. In Section 5, we outline our dynamic inference algorithm. In Section 6, we present the algorithms for dynamic Gibbs sampling. The conclusion is given in Section 7. The analyses of the dynamic sampling algorithms and the proof of the main theorem on dynamic inference are provided in the full version of the paper.

## 2 Dynamic inference problem

### 2.1 Markov random fields

An instance of (*pairwise*) *Markov random field (MRF)* is specified by a tuple  $\mathcal{I} = (V, E, Q, \Phi)$ , where  $G = (V, E)$  is an undirected simple graph;  $Q$  is a domain of  $q = |Q|$  *spin states*, for some finite  $q > 1$ ; and  $\Phi = (\phi_a)_{a \in V \cup E}$  associates each  $v \in V$  a *vertex potential*  $\phi_v : Q \rightarrow \mathbb{R}$  and each  $e \in E$  an *edge potential*  $\phi_e : Q^2 \rightarrow \mathbb{R}$ , where  $\phi_e$  is symmetric.

A *configuration*  $\sigma \in Q^V$  maps each vertex  $v \in V$  to a spin state in  $Q$ , so that each vertex can be interpreted as a variable. And the *Hamiltonian* of a configuration  $\sigma \in Q^V$  is defined as:

$$H(\sigma) \triangleq \sum_{v \in V} \phi_v(\sigma_v) + \sum_{e=\{u,v\} \in E} \phi_e(\sigma_u, \sigma_v).$$

This defines the *Gibbs distribution*  $\mu_{\mathcal{I}}$ , which is a probability distribution over  $Q^V$  such that

$$\forall \sigma \in Q^V, \quad \mu_{\mathcal{I}}(\sigma) = \frac{1}{Z} \exp(H(\sigma)),$$

where the normalizing factor  $Z \triangleq \sum_{\sigma \in Q^V} \exp(H(\sigma))$  is called the *partition function*.

The Gibbs measure  $\mu(\sigma)$  can be 0 as the functions  $\phi_v, \phi_e$  can take the value  $-\infty$ . A configuration  $\sigma$  is called *feasible* if  $\mu(\sigma) > 0$ . To trivialize the problem of constructing a feasible configuration, we further assume the following natural condition for the MRF instances considered in this paper:<sup>1</sup>

$$\forall v \in V, \forall \sigma \in Q^{\Gamma_G(v)} : \sum_{c \in Q} \exp \left( \phi_v(c) + \sum_{u \in \Gamma_v} \phi_{uv}(\sigma_u, c) \right) > 0. \quad (1)$$

where  $\Gamma_G(v) \triangleq \{u \in V \mid \{u, v\} \in E\}$  denotes the neighborhood of  $v$  in graph  $G = (V, E)$ .

Some well studied typical MRFs include:

- *Ising model*: The domain of each spin is  $Q = \{-1, +1\}$ . Each edge  $e \in E$  is associated with a *temperature*  $\beta_e \in \mathbb{R}$ ; and each vertex  $v \in V$  is associated with a *local field*  $h_v \in \mathbb{R}$ . For each configuration  $\sigma \in \{-1, +1\}^V$ ,  $\mu_{\mathcal{I}}(\sigma) \propto \exp \left( \sum_{\{u, v\} \in E} \beta_e \sigma_u \sigma_v + \sum_{v \in V} h_v \sigma_v \right)$ .
- *Hardcore model*: The domain is  $Q = \{0, 1\}$ . Each configuration  $\sigma \in Q^V$  indicates an independent set in  $G = (V, E)$ , and  $\mu_{\mathcal{I}}(\sigma) \propto \lambda^{|\sigma|}$ , where  $\lambda > 0$  is the *fugacity* parameter.
- *proper  $q$ -coloring*: uniform distribution over all proper  $q$ -colorings of  $G = (V, E)$ .

## 2.2 Probabilistic inference and sampling

In graphical models, the task of probabilistic inference is to derive the probabilities regarding one or more random variables of the model. Abstractly, this is described by a function  $\theta : \mathfrak{M} \rightarrow \mathbb{R}^K$  that maps each graphical model  $\mathcal{I} \in \mathfrak{M}$  to a target  $K$ -dimensional probability vector, where  $\mathfrak{M}$  is the class of graphical models containing the random variables we are interested in and the  $K$ -dimensional vector describes the probabilities we want to derive. Given  $\theta(\cdot)$  and an MRF instance  $\mathcal{I} \in \mathfrak{M}$ , the inference problem asks to estimate the probability vector  $\theta(\mathcal{I})$ .

Here are some fundamental inference problems [37] for MRF instances. Let  $\mathcal{I} = (V, E, Q, \Phi)$  be an MRF instance and  $A, B \subseteq V$  two disjoint sets where  $A \uplus B \subseteq V$ .

- *Marginal inference*: estimate the marginal distribution  $\mu_{A, \mathcal{I}}(\cdot)$  of the variables in  $A$ , where

$$\forall \sigma_A \in Q^A, \quad \mu_{A, \mathcal{I}}(\sigma_A) \triangleq \sum_{\tau \in Q^{V \setminus A}} \mu_{\mathcal{I}}(\sigma_A, \tau).$$

- *Posterior inference*: given any  $\tau_B \in Q^B$ , estimate the posterior distribution  $\mu_{A, \mathcal{I}}(\cdot \mid \tau_B)$  for the variables in  $A$ , where

$$\forall \sigma_A \in Q^A, \quad \mu_{A, \mathcal{I}}(\sigma_A \mid \tau_B) \triangleq \frac{\mu_{A \cup B, \mathcal{I}}(\sigma_A, \tau_B)}{\mu_{B, \mathcal{I}}(\tau_B)}.$$

- *Maximum-a-posteriori (MAP) inference*: find the maximum-a-posteriori (MAP) probabilities  $P_{A, \mathcal{I}}^*(\cdot)$  for the configurations over  $A$ , where

$$\forall \sigma_A \in Q^A, \quad P_{A, \mathcal{I}}^*(\sigma_A) \triangleq \max_{\tau_B \in Q^B} \mu_{A \cup B, \mathcal{I}}(\sigma_A, \tau_B).$$

<sup>1</sup> This condition guarantees that the marginal probabilities are always well-defined, and the problem of constructing a feasible configuration  $\sigma$ , where  $\mu_{\mathcal{I}}(\sigma) > 0$ , is trivial. The condition holds for all MRFs with soft constraints, or with hard constraints where there is a permissive spin, e.g. the hardcore model. For MRFs with truly repulsive hard constraints such as proper  $q$ -coloring, the condition may translate to the condition  $q \geq \Delta + 1$  where  $\Delta$  is the maximum degree of graph  $G$ , which is necessary for the irreducibility of local Markov chains for  $q$ -colorings.

All these fundamental inference problems can be described abstractly by a function  $\theta : \mathfrak{M} \rightarrow \mathbb{R}^K$ . For instances, for marginal inference,  $\mathfrak{M}$  contains all MRF instances where  $A$  is a subset of the vertices,  $K = |Q|^{|A|}$ , and  $\theta(\mathcal{I}) = (\mu_{A,\mathcal{I}}(\sigma_A))_{\sigma_A \in Q^A}$ ; and for posterior or MAP inference,  $\mathfrak{M}$  contains all MRF instances where  $A \uplus B$  is a subset of the vertices,  $K = |Q|^{|A|}$  and  $\theta(\mathcal{I}) = (\mu_{A,\mathcal{I}}(\sigma_A \mid \tau_B))_{\sigma_A \in Q^A}$  (for posterior inference) or  $\theta(\mathcal{I}) = (P_{A,\mathcal{I}}^*(\sigma_A))_{\sigma_A \in Q^A}$  (for MAP inference).

One canonical approach for probabilistic inference is by sampling: sufficiently many independent samples are drawn (approximately) from the Gibbs distribution of the MRF instance and an estimate of the target probabilities is calculated from these samples. Given a probabilistic inference problem  $\theta(\cdot)$ , we use  $\mathcal{E}_\theta(\cdot)$  to denote an estimating function that approximates  $\theta(\mathcal{I})$  using independent samples drawn approximately from  $\mu_{\mathcal{I}}$ . For the aforementioned problems of marginal, posterior and MAP inferences, such estimating function  $\mathcal{E}_\theta(\cdot)$  simply counts the frequency of the samples that satisfy certain properties.

The sampling cost of an estimator is captured in two aspects: the number of samples it uses and the accuracy of each individual sample it requires.

► **Definition 1** ( $(N, \epsilon)$ -estimator for  $\theta$ ). *Let  $\theta : \mathfrak{M} \rightarrow \mathbb{R}^K$  be a probabilistic inference problem and  $\mathcal{E}_\theta(\cdot)$  an estimating function for  $\theta(\cdot)$  that for each instance  $\mathcal{I} = (V, E, Q, \Phi) \in \mathfrak{M}$ , maps samples in  $Q^V$  to an estimate of  $\theta(\mathcal{I})$ . Let  $N : \mathbb{N}^+ \rightarrow \mathbb{N}^+$  and  $\epsilon : \mathbb{N}^+ \rightarrow (0, 1)$ . For any instance  $\mathcal{I} = (V, E, Q, \Phi) \in \mathfrak{M}$  where  $n = |V|$ , the random variable  $\mathcal{E}_\theta(\mathbf{X}^{(1)}, \dots, \mathbf{X}^{(N(n))})$  is said to be an  $(N, \epsilon)$ -estimator for  $\theta(\mathcal{I})$  if  $\mathbf{X}^{(1)}, \dots, \mathbf{X}^{(N(n))} \in Q^V$  are  $N(n)$  independent samples drawn approximately from  $\mu_{\mathcal{I}}$  such that  $d_{\text{TV}}(\mathbf{X}^{(j)}, \mu_{\mathcal{I}}) \leq \epsilon(n)$  for all  $1 \leq j \leq N(n)$ .*

In Definition 1, an estimator is viewed as a black-box algorithm specified by two functions  $N$  and  $\epsilon$ . Usually, the estimator is more accurate if more independent samples are drawn and each sample provides a higher level of accuracy. Thus, one can choose some large  $N$  and small  $\epsilon$  to achieve a desired quality of estimate.

## 2.3 Dynamic inference problem

We consider the inference problem where the input graphical model is changed dynamically: at each step, the current MRF instance  $\mathcal{I} = (V, E, Q, \Phi)$  is updated to a new instance  $\mathcal{I}' = (V', E', Q, \Phi')$ . We consider general update operations for MRFs that can change both the **underlying graph** and **all edge/vertex potentials** simultaneously, where the update request is made by a *non-adaptive adversary* independently of the randomness used by the inference algorithm. Such updates are general enough and cover many applications, e.g. analyses of time series network data [5, 32, 14, 2], and learning algorithms for graphical models [22, 37].

The difference between the original and the updated instances is measured as follows.

► **Definition 2** (difference between MRF instances). *The difference between two MRF instances  $\mathcal{I} = (V, E, Q, \Phi)$  and  $\mathcal{I}' = (V', E', Q, \Phi')$ , where  $\Phi = (\phi_a)_{a \in V \cup E}$  and  $\Phi' = (\phi'_a)_{a \in V' \cup E'}$ , is defined as*

$$d(\mathcal{I}, \mathcal{I}') \triangleq \sum_{v \in V \cap V'} \|\phi_v - \phi'_v\|_1 + \sum_{e \in E \cap E'} \|\phi_e - \phi'_e\|_1 + |V \oplus V'| + |E \oplus E'|, \quad (2)$$

where  $A \oplus B = (A \setminus B) \cup (B \setminus A)$  stands for the symmetric difference between two sets  $A$  and  $B$ ,  $\|\phi_v - \phi'_v\|_1 \triangleq \sum_{c \in Q} |\phi_v(c) - \phi'_v(c)|$ , and  $\|\phi_e - \phi'_e\|_1 \triangleq \sum_{c, c' \in Q} |\phi_e(c, c') - \phi'_e(c, c')|$ .



Given a probability vector specified by the function  $\theta : \mathfrak{M} \rightarrow \mathbb{R}^K$ , the *dynamic inference problem* asks to maintain an estimator  $\hat{\theta}(\mathcal{I})$  of  $\theta(\mathcal{I})$  for the current MRF instance  $\mathcal{I} = (V, E, Q, \Phi) \in \mathfrak{M}$ , with a data structure, such that when  $\mathcal{I}$  is updated to  $\mathcal{I}' = (V', E', Q, \Phi') \in \mathfrak{M}$ , the algorithm updates  $\hat{\theta}(\mathcal{I})$  to an estimator  $\hat{\theta}(\mathcal{I}')$  of the new vector  $\theta(\mathcal{I}')$ , or equivalently, outputs the difference between the estimators  $\hat{\theta}(\mathcal{I})$  and  $\hat{\theta}(\mathcal{I}')$ .

It is desirable to have the dynamic inference algorithm which maintains an  $(N, \epsilon)$ -estimator for  $\theta(\mathcal{I})$  for the current instance  $\mathcal{I}$ . However, the dynamic algorithm cannot be efficient if  $N(n)$  and  $\epsilon(n)$  change drastically with  $n$  (so that significantly more samples or substantially more accurate samples may be needed when a new vertex is added), or if recalculating the estimating function  $\mathcal{E}_\theta(\cdot)$  itself is expensive. We introduce a notion of *dynamical efficiency* for the estimators that are suitable for dynamic inference.

► **Definition 3** (dynamical efficiency). *Let  $N : \mathbb{N}^+ \rightarrow \mathbb{N}^+$  and  $\epsilon : \mathbb{N}^+ \rightarrow (0, 1)$ . Let  $\mathcal{E}(\cdot)$  be an estimating function for some  $K$ -dimensional probability vector of MRF instances. A tuple  $(N, \epsilon, \mathcal{E})$  is said to be dynamically efficient if it satisfies:*

■ **(bounded difference)** *there exist constants  $C_1, C_2 > 0$  such that for any  $n \in \mathbb{N}^+$ ,*

$$|N(n+1) - N(n)| \leq \frac{C_1 \cdot N(n)}{n} \quad \text{and} \quad |\epsilon(n+1) - \epsilon(n)| \leq \frac{C_2 \cdot \epsilon(n)}{n};$$

■ **(small incremental cost)** *there is a deterministic algorithm that maintains  $\mathcal{E}(\mathbf{X}^{(1)}, \dots, \mathbf{X}^{(m)})$  using  $(mn + K) \cdot \text{polylog}(mn)$  bits where  $\mathbf{X}^{(1)}, \dots, \mathbf{X}^{(m)} \in Q^V$  and  $n = |V|$ , such that when  $\mathbf{X}^{(1)}, \dots, \mathbf{X}^{(m)} \in Q^V$  are updated to  $\mathbf{Y}^{(1)}, \dots, \mathbf{Y}^{(m')} \in Q^{V'}$ , where  $n' = |V'|$ , the algorithm updates  $\mathcal{E}(\mathbf{X}^{(1)}, \dots, \mathbf{X}^{(m)})$  to  $\mathcal{E}(\mathbf{Y}^{(1)}, \dots, \mathbf{Y}^{(m')})$  within time cost  $\mathcal{D} \cdot \text{polylog}(mm'nn') + O(m + m')$ , where  $\mathcal{D}$  is the size of the difference between two sample sequences defined as:*

$$\mathcal{D} \triangleq \sum_{i \leq \max\{m, m'\}} \sum_{v \in V \cup V'} \mathbf{1}[\mathbf{X}^{(i)}(v) \neq \mathbf{Y}^{(i)}(v)], \quad (3)$$

where an unassigned  $\mathbf{X}^{(i)}(v)$  or  $\mathbf{Y}^{(i)}(v)$  is not equal to any assigned spin.

The dynamic efficiency basically asks  $N(\cdot), \epsilon(\cdot)$ , and  $\mathcal{E}(\cdot)$  to have some sort of “Lipschitz” properties. To satisfy the bounded difference condition,  $N(n)$  and  $1/\epsilon(n)$  are necessarily polynomially bounded, and they can be any constant, polylogarithmic, or polynomial functions, or multiplications of such functions. The condition with small incremental cost also holds very commonly. In particular, it is satisfied by the estimating functions for all the aforementioned problems for the marginal, posterior and MAP inferences as long as the sets of variables have sizes  $|A|, |B| = O(\log n)$ . We remark that the  $O(\log n)$  upper bound is somehow necessary for the efficiency of inference, because otherwise the dimension of  $\theta(\mathcal{I})$  itself (which is at least  $q^{|A|}$ ) becomes super-polynomial in  $n$ .

### 3 Main results

Let  $\mathcal{I} = (V, E, Q, \Phi)$  be an MRF instance, where  $G = (V, E)$ . Let  $\Gamma_G(v)$  denote the neighborhood of  $v$  in  $G$ . For any vertex  $v \in V$  and any configuration  $\sigma \in Q^{\Gamma_G(v)}$ , we use  $\mu_{v, \mathcal{I}}^\sigma(\cdot) = \mu_{v, \mathcal{I}}(\cdot \mid \sigma)$  to denote the marginal distribution on  $v$  conditional on  $\sigma$ :

$$\forall c \in Q : \quad \mu_{v, \mathcal{I}}^\sigma(c) = \mu_{v, \mathcal{I}}(c \mid \sigma) \triangleq \frac{\exp\left(\phi_v(c) + \sum_{u \in \Gamma_G(v)} \phi_{uv}(\sigma_u, c)\right)}{\sum_{a \in Q} \exp\left(\phi_v(a) + \sum_{u \in \Gamma_G(v)} \phi_{uv}(\sigma_u, a)\right)}.$$

Due to the assumption in (1), the marginal distribution is always well-defined. The following condition is the *Dobrushin-Shlosman condition* [8, 9, 7, 19, 12].



► **Condition 4** (Dobrushin-Shlosman condition). Let  $\mathcal{I} = (V, E, Q, \Phi)$  be an MRF instance with Gibbs distribution  $\mu = \mu_{\mathcal{I}}$ . Let  $A_{\mathcal{I}} \in \mathbb{R}_{\geq 0}^{V \times V}$  be the influence matrix which is defined as

$$A_{\mathcal{I}}(u, v) \triangleq \begin{cases} \max_{(\sigma, \tau) \in B_{u,v}} d_{\text{TV}}(\mu_v^{\sigma}, \mu_v^{\tau}), & \{u, v\} \in E, \\ 0 & \{u, v\} \notin E, \end{cases}$$

where the maximum is taken over the set  $B_{u,v}$  of all  $(\sigma, \tau) \in Q^{\Gamma_G(v)} \times Q^{\Gamma_G(v)}$  that differ only at  $u$ , and  $d_{\text{TV}}(\mu_v^{\sigma}, \mu_v^{\tau}) \triangleq \frac{1}{2} \sum_{c \in Q} |\mu_v^{\sigma}(c) - \mu_v^{\tau}(c)|$  is the total variation distance between  $\mu_v^{\sigma}$  and  $\mu_v^{\tau}$ . An MRF instance  $\mathcal{I}$  is said to satisfy the Dobrushin-Shlosman condition if there is a constant  $\delta > 0$  such that

$$\max_{u \in V} \sum_{v \in V} A_{\mathcal{I}}(u, v) \leq 1 - \delta.$$

Our main theorem assumes the following setup: Let  $\theta : \mathfrak{M} \rightarrow \mathbb{R}^K$  be a probabilistic inference problem that maps each MRF instance in  $\mathfrak{M}$  to a  $K$ -dimensional probability vector, and let  $\mathcal{E}_{\theta}$  be its estimating function. Let  $N : \mathbb{N}^+ \rightarrow \mathbb{N}^+$  and  $\epsilon : \mathbb{N}^+ \rightarrow (0, 1)$ . We use  $\mathcal{I} = (V, E, Q, \Phi) \in \mathfrak{M}$ , where  $n = |V|$ , to denote the current instance and  $\mathcal{I}' = (V', E', Q, \Phi') \in \mathfrak{M}$ , where  $n' = |V'|$ , to denote the updated instance.

► **Theorem 5** (dynamic inference algorithm). Assume that  $(N, \epsilon, \mathcal{E}_{\theta})$  is dynamically efficient, both  $\mathcal{I}$  and  $\mathcal{I}'$  satisfy the Dobrushin-Shlosman condition, and  $d(\mathcal{I}, \mathcal{I}') \leq L = o(n)$ .

There is an algorithm that maintains an  $(N, \epsilon)$ -estimator  $\hat{\theta}(\mathcal{I})$  of the probability vector  $\theta(\mathcal{I})$  for the current MRF instance  $\mathcal{I}$ , using  $\tilde{O}(nN(n) + K)$  bits, such that when  $\mathcal{I}$  is updated to  $\mathcal{I}'$ , the algorithm updates  $\hat{\theta}(\mathcal{I})$  to an  $(N, \epsilon)$ -estimator  $\hat{\theta}(\mathcal{I}')$  of  $\theta(\mathcal{I}')$  for the new instance  $\mathcal{I}'$ , within expected time cost

$$\tilde{O}(\Delta^2 LN(n) + \Delta n),$$

where  $\tilde{O}(\cdot)$  hides a  $\text{polylog}(n)$  factor,  $\Delta = \max\{\Delta_G, \Delta_{G'}\}$ , where  $\Delta_G$  and  $\Delta_{G'}$  denote the maximum degree of  $G = (V, E)$  and  $G' = (V', E')$  respectively.

Note that the extra  $O(\Delta n)$  cost is necessary for editing the current MRF instance  $\mathcal{I}$  to  $\mathcal{I}'$ .

Typically, the difference between two MRF instances  $\mathcal{I}, \mathcal{I}'$  is small<sup>2</sup>, and the underlying graphs are sparse [6], that is,  $L, \Delta \leq \text{polylog}(n)$ . In such cases, our algorithm updates the estimator within time cost  $\tilde{O}(N(n) + n)$ , which significantly outperforms static sampling-based inference algorithms that require time cost  $\Omega(n'N(n')) = \Omega(nN(n))$  for redrawing all  $N(n')$  independent samples.

**Dynamic sampling.** The core of our dynamic inference algorithm is a dynamic sampling algorithm: Assuming the Dobrushin-Shlosman condition, the algorithm can maintain a sequence of  $N(n)$  independent samples  $\mathbf{X}^{(1)}, \dots, \mathbf{X}^{(N(n))} \in Q^V$  that are  $\epsilon(n)$ -close to  $\mu_{\mathcal{I}}$  in total variation distance, and when  $\mathcal{I}$  is updated to  $\mathcal{I}'$  with difference  $d(\mathcal{I}, \mathcal{I}') \leq L = o(n)$ , the algorithm can update the maintained samples to  $N(n')$  independent samples  $\mathbf{Y}^{(1)}, \dots, \mathbf{Y}^{(N(n'))} \in Q^{V'}$  that are  $\epsilon(n')$ -close to  $\mu_{\mathcal{I}'}$  in total variation distance, using a time cost  $\tilde{O}(\Delta^2 LN(n) + \Delta n)$  in expectation. This shows an “algorithmic Lipschitz” condition

<sup>2</sup> In multivariate time-series data analysis, the MRF instances of two sequential times are similar. In the iterative algorithms for learning graphical models, the difference between two sequential MRF instances generated by gradient descent are bounded to prevent oscillations. Specifically, the difference is very small when the iterative algorithm approaches to the convergence state [22, 37].

holds for sampling from Gibbs distributions: when the MRF changes insignificantly, a population of samples can be modified to reflect the new distribution, with cost proportional to the difference on MRF. We show that such property is guaranteed by the Dobrushin-Shlosman condition. This dynamic sampling algorithm is formally described in Theorem 9 and is of independent interest [14].

**Applications on specific models.** On specific models, we have the following results, where  $\delta > 0$  is an arbitrary constant.

■ **Table 1** Dynamic inference for specific models.

model	regime	space cost	time cost for each update
Ising	$e^{-2 \beta } \geq 1 - \frac{2-\delta}{\Delta+1}$	$\tilde{O}(nN(n) + K)$	$\tilde{O}(\Delta^2 LN(n) + \Delta n)$
hardcore	$\lambda \leq \frac{2-\delta}{\Delta-2}$	$\tilde{O}(nN(n) + K)$	$\tilde{O}(\Delta^3 LN(n) + \Delta n)$
$q$ -coloring	$q \geq (2 + \delta)\Delta$	$\tilde{O}(nN(n) + K)$	$\tilde{O}(\Delta^2 LN(n) + \Delta n)$

The results for Ising model and  $q$ -coloring are corollaries of Theorem 5. The regime for hardcore model is better than the Dobrushin-Shlosman condition (which is  $\lambda \leq \frac{1-\delta}{\Delta-1}$ ), because we use the coupling introduced by Vigoda [36] to analyze the algorithm.

## 4 Preliminaries

**Total variation distance and coupling.** Let  $\mu$  and  $\nu$  be two distributions over  $\Omega$ . The *total variation distance* between  $\mu$  and  $\nu$  is defined as

$$d_{\text{TV}}(\mu, \nu) \triangleq \frac{1}{2} \sum_{x \in \Omega} |\mu(x) - \nu(x)|.$$

A *coupling* of  $\mu$  and  $\nu$  is a joint distribution  $(X, Y) \in \Omega \times \Omega$  such that marginal distribution of  $X$  is  $\mu$  and the marginal distribution of  $Y$  is  $\nu$ . The following coupling lemma is well-known.

► **Proposition 6** (coupling lemma). *For any coupling  $(X, Y)$  of  $\mu$  and  $\nu$ , it holds that*

$$\Pr[X \neq Y] \geq d_{\text{TV}}(\mu, \nu).$$

*Furthermore, there is an optimal coupling that achieves equality.*

**Local neighborhood.** Let  $G = (V, E)$  be a graph. For any vertex  $v \in V$ , let  $\Gamma_G(v) \triangleq \{u \in V \mid \{u, v\} \in E\}$  denote the neighborhood of  $v$ , and  $\Gamma_G^+(v) \triangleq \Gamma_G(v) \cup \{v\}$  the inclusive neighborhood of  $v$ . We simply write  $\Gamma_v = \Gamma(v) = \Gamma_G(v)$  and  $\Gamma_v^+ = \Gamma^+(v) = \Gamma_G^+(v)$  for short when  $G$  is clear in the context. We use  $\Delta = \Delta_G \triangleq \max_{v \in V} |\Gamma_v|$  to denote the maximum degree of graph  $G$ .

A notion of **local neighborhood for MRF** is frequently used. Let  $\mathcal{I} = (V, E, Q, \Phi)$  be an MRF instance. For  $v \in V$ , we denote by  $\mathcal{I}_v \triangleq \mathcal{I}[\Gamma_v^+]$  the restriction of  $\mathcal{I}$  on the inclusive neighborhood  $\Gamma_v^+$  of  $v$ , i.e.  $\mathcal{I}_v = (\Gamma_v^+, E_v, Q, \Phi_v)$ , where  $E_v = \{\{u, v\} \in E\}$  and  $\Phi_v = (\phi_a)_{a \in \Gamma_v^+ \cup E_v}$ .

**Gibbs sampling.** The *Gibbs sampling* (a.k.a. *heat-bath*, *Glauber dynamics*), is a classic Markov chain for sampling from Gibbs distributions. Let  $\mathcal{I} = (V, E, Q, \Phi)$  be an MRF instance and  $\mu = \mu_{\mathcal{I}}$  its Gibbs distribution. The chain of Gibbs sampling (Algorithm 1) is on the space  $\Omega \triangleq Q^V$ , and has the stationary distribution  $\mu_{\mathcal{I}}$  [27, Chapter 3].

■ **Algorithm 1** Gibbs sampling.

---

**Initialization:** an initial state  $\mathbf{X}_0 \in \Omega$  (not necessarily feasible);

- 1 **for**  $t = 1, 2, \dots, T$  **do**
- 2     pick  $v_t \in V$  uniformly at random;
- 3     draw a random value  $c \in Q$  from the marginal distribution  $\mu_{v_t}(\cdot \mid X_{t-1}(\Gamma_{v_t}))$ ;
- 4      $X_t(v_t) \leftarrow c$  and  $X_t(u) \leftarrow X_{t-1}(u)$  for all  $u \in V \setminus \{v_t\}$ ;

---

**Marginal distributions.** Here  $\mu_v(\cdot \mid \sigma(\Gamma_v)) = \mu_{v, \mathcal{I}}(\cdot \mid \sigma(\Gamma_v))$  denotes the marginal distribution at  $v \in V$  conditioning on  $\sigma(\Gamma_v) \in Q^{\Gamma_v}$ , which is computed as:

$$\forall c \in Q : \quad \mu_v(c \mid \sigma(\Gamma_v)) = \frac{\phi_v(c) \prod_{u \in \Gamma_v} \phi_{uv}(\sigma_u, c)}{\sum_{c' \in Q} \phi_v(c') \prod_{u \in \Gamma_v} \phi_{uv}(\sigma_u, c')}. \quad (4)$$

Due to the assumption (1), this marginal distribution is always well defined, and its computation uses only the information of  $\mathcal{I}_v$ .

**Coupling for mixing time.** Consider a chain  $(\mathbf{X}_t)_{t=0}^{\infty}$  on space  $\Omega$  with stationary distribution  $\mu_{\mathcal{I}}$  for MRF instance  $\mathcal{I}$ . The *mixing rate* is defined as: for  $\epsilon > 0$ ,

$$\tau_{\text{mix}}(\mathcal{I}, \epsilon) \triangleq \max_{\mathbf{X}_0} \min \{t \mid d_{\text{TV}}(\mathbf{X}_t, \mu_{\mathcal{I}}) \leq \epsilon\},$$

where  $d_{\text{TV}}(\mathbf{X}_t, \mu_{\mathcal{I}})$  denotes the *total variation distance* between  $\mu_{\mathcal{I}}$  and the distribution of  $\mathbf{X}_t$ .

A coupling of a Markov chain is a joint process  $(\mathbf{X}_t, \mathbf{Y}_t)_{t \geq 0}$  such that  $(\mathbf{X}_t)_{t \geq 0}$  and  $(\mathbf{Y}_t)_{t \geq 0}$  marginally follow the same transition rule as the original chain. Consider the following type of couplings.

► **Definition 7** (one-step optimal coupling for Gibbs sampling). A coupling  $(\mathbf{X}_t, \mathbf{Y}_t)_{t \geq 0}$  of Gibbs sampling on an MRF instance  $\mathcal{I} = (V, E, Q, \Phi)$  is a one-step optimal coupling if it is constructed as follows: For  $t = 1, 2, \dots$ ,

1. pick the same random  $v_t \in V$ , and let  $(X_t(u), Y_t(u)) \leftarrow (X_{t-1}(u), Y_{t-1}(u))$  for all  $u \neq v_t$ ;
2. sample  $(X_t(v_t), Y_t(v_t))$  from an optimal coupling  $D_{\text{opt}, \mathcal{I}_{v_t}}^{\sigma, \tau}(\cdot, \cdot)$  of the marginal distributions  $\mu_{v_t}(\cdot \mid \sigma)$  and  $\mu_{v_t}(\cdot \mid \tau)$  where  $\sigma = X_{t-1}(\Gamma_{v_t})$  and  $\tau = Y_{t-1}(\Gamma_{v_t})$ .

The coupling  $D_{\text{opt}, \mathcal{I}_{v_t}}^{\sigma, \tau}(\cdot, \cdot)$  is an *optimal coupling* of  $\mu_{v_t}(\cdot \mid \sigma)$  and  $\mu_{v_t}(\cdot \mid \tau)$  that attains the maximum  $\Pr[\mathbf{x} = \mathbf{y}]$  for all couplings  $(\mathbf{x}, \mathbf{y})$  of  $\mathbf{x} \sim \mu_{v_t}(\cdot \mid \sigma)$  and  $\mathbf{y} \sim \mu_{v_t}(\cdot \mid \tau)$ . The coupling  $D_{\text{opt}, \mathcal{I}_{v_t}}^{\sigma, \tau}(\cdot, \cdot)$  is determined by the local information  $\mathcal{I}_v$  and  $\sigma, \tau \in Q^{\deg(v)}$ .

With such a coupling, we can establish the following relation between the Dobrushin-Shlosman condition and the rapid mixing of the Gibbs sampling [8, 9, 7, 4, 19, 12].

► **Proposition 8** ([4, 19]). Let  $\mathcal{I} = (V, E, Q, \Phi)$  be an MRF instance with  $n = |V|$ , and  $\Omega = Q^V$  the state space. Let  $H(\sigma, \tau) \triangleq |\{v \in V \mid \sigma_v \neq \tau_v\}|$  denote the Hamming distance between  $\sigma \in \Omega$  and  $\tau \in \Omega$ . If  $\mathcal{I}$  satisfies the Dobrushin-Shlosman condition (Condition 4) with constant  $\delta > 0$ , then the one-step optimal coupling  $(\mathbf{X}_t, \mathbf{Y}_t)_{t \geq 0}$  for Gibbs sampling (Definition 7) satisfies

$$\forall \sigma, \tau \in \Omega : \quad \mathbb{E}[H(\mathbf{X}_t, \mathbf{Y}_t) \mid \mathbf{X}_{t-1} = \sigma \wedge \mathbf{Y}_{t-1} = \tau] \leq \left(1 - \frac{\delta}{n}\right) \cdot H(\sigma, \tau),$$

and hence the mixing rate of Gibbs sampling on  $\mathcal{I}$  is bounded as  $\tau_{\text{mix}}(\mathcal{I}, \epsilon) \leq \lceil \frac{n}{\delta} \log \frac{n}{\epsilon} \rceil$ .

## 5 Outlines of algorithm

Let  $\theta : \mathfrak{M} \rightarrow \mathbb{R}^K$  be a probabilistic inference problem that maps each MRF instance in  $\mathfrak{M}$  to a  $K$ -dimensional probability vector, and let  $\mathcal{E}_\theta$  be its estimating function. Let  $\mathcal{I} = (V, E, Q, \Phi) \in \mathfrak{M}$  be the current instance, where  $n = |V|$ . Our dynamic inference algorithm maintains a sequence of  $N(n)$  independent samples  $\mathbf{X}^{(1)}, \dots, \mathbf{X}^{(N(n))} \in Q^V$  which are  $\epsilon(n)$ -close to the Gibbs distribution  $\mu_{\mathcal{I}}$  in total variation distance and an  $(N, \epsilon)$ -estimator  $\hat{\theta}(\mathcal{I})$  of  $\theta(\mathcal{I})$  such that

$$\hat{\theta}(\mathcal{I}) = \mathcal{E}_\theta(\mathbf{X}^{(1)}, \mathbf{X}^{(2)}, \dots, \mathbf{X}^{(N(n))}).$$

Upon an update request that modifies  $\mathcal{I}$  to a new instance  $\mathcal{I}' = (V', E', Q, \Phi') \in \mathfrak{M}$ , where  $n' = |V'|$ , our algorithm does the followings:

- *Update the sample sequence.* Update  $\mathbf{X}^{(1)}, \dots, \mathbf{X}^{(N(n))}$  to a new sequence of  $N(n')$  independent samples  $\mathbf{Y}^{(1)}, \dots, \mathbf{Y}^{(N(n'))} \in Q^{V'}$  that are  $\epsilon(n')$ -close to  $\mu_{\mathcal{I}'}$  in total variation distance, and output the difference between two sample sequences.
- *Update the estimator.* Given the difference between the two sample sequences, update  $\hat{\theta}(\mathcal{I})$  to  $\hat{\theta}(\mathcal{I}') = \mathcal{E}_\theta(\mathbf{Y}^{(1)}, \dots, \mathbf{Y}^{(N(n'))})$  by accessing the oracle in Definition 3.

Obviously, the updated estimator  $\hat{\theta}(\mathcal{I}')$  is an  $(N, \epsilon)$ -estimator for  $\theta(\mathcal{I}')$ .

Our main technical contribution is to give an algorithm that dynamically maintains a sequence of  $N(n)$  independent samples for  $\mu_{\mathcal{I}}$ , while  $\mathcal{I}$  itself is dynamically changing. The dynamic sampling problem was recently introduced in [14]. The dynamical sampling algorithm given there only handles update of a single vertex or edge and works only for graphical models with soft constraints.

In contrast, our dynamic sampling algorithm maintains a sequence of  $N(n)$  independent samples for  $\mu_{\mathcal{I}}$  within total variation distance  $\epsilon(n)$ , while the entire specification of the graphical model  $\mathcal{I}$  is subject to dynamic update (to a new  $\mathcal{I}'$  with difference  $d(\mathcal{I}, \mathcal{I}') \leq L = o(n)$ ). Specifically, the algorithm updates the sample sequence within expected time  $O(\Delta^2 N(n) L \log^3 n + \Delta n)$ . Note that the extra  $O(\Delta n)$  cost is necessary for just editing the current MRF instance  $\mathcal{I}$  to  $\mathcal{I}'$  because a single update may change all the vertex and edge potentials simultaneously. This incremental time cost dominates the time cost of the dynamic inference algorithm, and is efficient for maintaining  $N(n)$  independent samples, especially when  $N(n)$  is sufficiently large, e.g.  $N(n) = \Omega(n/L)$ , in which case the average incremental cost for updating each sample is  $O(\Delta^2 L \log^3 n + \Delta n/N(n)) = O(\Delta^2 L \log^3 n)$ .

We illustrate the main idea by explaining how to maintain one sample. The idea is to represent the trace of the Markov chain for generating the sample by a dynamic data structure, and when the MRF instance is changed, this trace is modified to that of the new chain for generating the sample for the updated instance. This is achieved by both a set of efficient dynamic data structures and the coupling between the two Markov chains.

Specifically, let  $(\mathbf{X}_t)_{t=0}^T$  be the Gibbs sampler chain for distribution  $\mu_{\mathcal{I}}$ . When the chain is rapidly mixing, starting from an arbitrary initial configuration  $\mathbf{X}_0 \in Q^V$ , after suitably many steps  $\mathbf{X} = \mathbf{X}_T$  is an accurate enough sample for  $\mu_{\mathcal{I}}$ . At each step,  $\mathbf{X}_{t-1}$  and  $\mathbf{X}_t$  may differ only at a vertex  $v_t$  which is picked from  $V$  uniformly and independently at random. The evolution of the chain is fully captured by the initial state  $\mathbf{X}_0$  and the sequence of pairs  $\langle v_t, X_t(v_t) \rangle$ , from  $t = 1$  to  $t = T$ , which is called the *execution log* of the chain in the paper.

Now suppose that the current instance  $\mathcal{I}$  is updated to  $\mathcal{I}'$ . We construct such a coupling between the original chain  $(\mathbf{X}_t)_{t=0}^T$  and the new chain  $(\mathbf{Y}_t)_{t=0}^T$ , such that  $(\mathbf{Y}_t)_{t=0}^T$  is a faithful Gibbs sampling chain for the updated instance  $\mathcal{I}'$  given that  $(\mathbf{X}_t)_{t=0}^T$  is a faithful chain for  $\mathcal{I}$ , and the difference between the two chains is small, in the sense that they have almost the same execution logs except for about  $O(TL/n)$  steps, where  $L$  is the difference between  $\mathcal{I}$  and  $\mathcal{I}'$ .

To simplify the exposition of such coupling, for now we restrict ourselves to the cases where the update to the instance  $\mathcal{I}$  does not change the set of variables. Without loss of generality, we only consider the following two basic update operations that modifies  $\mathcal{I}$  to  $\mathcal{I}'$ .

- *Graph update.* The update only adds or deletes some edges, while all vertex potentials and the potentials of unaffected edges are not changed.
- *Hamiltonian update.* The update changes (possibly all) potentials of vertices and edges, while the underlying graph remains unchanged.

The general update of graphical model can be obtained by combining these two basic operations.

Then the new chain  $(\mathbf{Y}_t)_{t=0}^T$  can be coupled with  $(\mathbf{X}_t)_{t=0}^T$  by using the same initial configuration  $\mathbf{Y}_0 = \mathbf{X}_0$  and the same sequence  $v_1, v_2, \dots, v_T \in V$  of randomly picked vertices. And for  $t = 1, 2, \dots, T$ , the transition  $\langle v_t, Y_t(v_t) \rangle$  of the new chain can be generated using the same vertex  $v_t$  as in the original  $(\mathbf{X}_t)_{t=0}^T$  chain, and a random  $Y_t(v_t)$  generated according to a coupling of the marginal distributions of  $X_t(v_t)$  and  $Y_t(v_t)$ , conditioning respectively on the current states of the neighborhood of  $v_t$  in  $(\mathbf{X}_t)_{t=0}^T$  and  $(\mathbf{Y}_t)_{t=0}^T$ . Note that these two marginal distributions must be identical unless (I)  $\mathbf{X}_{t-1}$  and  $\mathbf{Y}_{t-1}$  differ from each other over the neighborhood of  $v_t$  or (II) the  $v_t$  itself is incident to where the models  $\mathcal{I}$  and  $\mathcal{I}'$  differ. The event (II) occurs rarely due to the following reasons.

- For graph update, the event (II) occurs only if  $v_t$  is incident to an updated edge. Since only  $L$  edges are updated, the event occurs in at most  $O(TL/n)$  steps in expectation.
- For Hamiltonian update, all the potentials of vertices and edges can be changed, thus  $\mathcal{I}, \mathcal{I}'$  may differ everywhere. The key observation is that, as the total difference between the current and updated potentials is bounded by  $L$ , we can apply a filter to first select all candidate steps where the coupling may actually fail due to the difference between  $\mathcal{I}$  and  $\mathcal{I}'$ , which can be as small as  $O(TL/n)$ , and the actual coupling between  $(\mathbf{X}_t)_{t=0}^\infty$  and  $(\mathbf{Y}_t)_{t=0}^\infty$  is constructed with such prior.

Finally, when  $\mathcal{I}$  and  $\mathcal{I}'$  both satisfy the Dobrushin-Shlosman condition, the percolation of disagreements between  $(\mathbf{X}_t)_{t=0}^T$  and  $(\mathbf{Y}_t)_{t=0}^T$  is bounded, and we show that the two chains are almost always identically coupled as  $\langle v_t, X_t(v_t) \rangle = \langle v_t, Y_t(v_t) \rangle$ , with exceptions at only  $O(TL/n)$  steps. The original chain  $(\mathbf{X}_t)_{t=0}^T$  can then be updated to the new chain  $(\mathbf{Y}_t)_{t=0}^T$  by only editing these  $O(TL/n)$  local transitions  $\langle v_t, Y_t(v_t) \rangle$  which are different from  $\langle v_t, X_t(v_t) \rangle$ . This is aided by the dynamic data structure for the execution log of the chain, which is of independent interest.

## 6 Dynamic Gibbs sampling

In this section, we give the dynamic sampling algorithm that updates the sample sequences.

In the following theorem, we use  $\mathcal{I} = (V, E, Q, \Phi)$ , where  $n = |V|$ , to denote the current MRF instance and  $\mathcal{I}' = (V', E', Q, \Phi')$ , where  $n' = |V'|$ , to denote the updated MRF instance. And define

$$\begin{aligned} d_{\text{graph}}(\mathcal{I}, \mathcal{I}') &\triangleq |V \oplus V'| + |E \oplus E'| \\ d_{\text{Hamil}}(\mathcal{I}, \mathcal{I}') &\triangleq \sum_{v \in V \cap V'} \|\phi_v - \phi'_v\|_1 + \sum_{e \in E \cap E'} \|\phi_e - \phi'_e\|_1. \end{aligned}$$

Note that  $d(\mathcal{I}, \mathcal{I}') = d_{\text{graph}}(\mathcal{I}, \mathcal{I}') + d_{\text{Hamil}}(\mathcal{I}, \mathcal{I}')$ , where  $d(\mathcal{I}, \mathcal{I}')$  is defined in (2).

► **Theorem 9** (dynamic sampling algorithm). *Let  $N : \mathbb{N}^+ \rightarrow \mathbb{N}^+$  and  $\epsilon : \mathbb{N}^+ \rightarrow (0, 1)$  be two functions satisfying the bounded difference condition in Definition 3. Assume that  $\mathcal{I}$  and  $\mathcal{I}'$  both satisfy Dobrushin-Shlosman condition,  $d_{\text{graph}}(\mathcal{I}, \mathcal{I}') \leq L_{\text{graph}} = o(n)$  and  $d_{\text{Hamil}}(\mathcal{I}, \mathcal{I}') \leq L_{\text{Hamil}}$ .*

*There is an algorithm that maintains a sequence of  $N(n)$  independent samples  $\mathbf{X}^{(1)}, \dots, \mathbf{X}^{(N(n))} \in Q^V$  where  $d_{\text{TV}}(\mu_{\mathcal{I}}, \mathbf{X}^{(i)}) \leq \epsilon(n)$  for all  $1 \leq i \leq N(n)$ , using  $O(nN(n) \log n)$  memory words, each of  $O(\log n)$  bits, such that when  $\mathcal{I}$  is updated to  $\mathcal{I}'$ , the algorithm updates the sequence to  $N(n')$  independent samples  $\mathbf{Y}^{(1)}, \dots, \mathbf{Y}^{(N(n'))} \in Q^{V'}$  where  $d_{\text{TV}}(\mu_{\mathcal{I}'}, \mathbf{Y}^{(i)}) \leq \epsilon(n')$  for all  $1 \leq i \leq N(n')$ , within expected time cost*

$$O(\Delta^2(L_{\text{graph}} + L_{\text{Hamil}})N(n) \log^3 n + \Delta n), \quad (5)$$

where  $\Delta = \max\{\Delta_G, \Delta_{G'}\}$ , and  $\Delta_G, \Delta_{G'}$  denote the maximum degree of  $G = (V, E)$  and  $G' = (V', E')$ .

Our algorithm is based on the Gibbs sampling algorithm. Let  $N : \mathbb{N}^+ \rightarrow \mathbb{N}^+$  and  $\epsilon : \mathbb{N}^+ \rightarrow (0, 1)$  be two functions in Theorem 9. We first give the *single-sample dynamic Gibbs sampling algorithm* (Algorithm 2) that maintains a single sample  $\mathbf{X} \in Q^V$  for the current MRF instance  $\mathcal{I} = (V, E, Q, \Phi)$  where  $n = |V|$  such that  $d_{\text{TV}}(\mathbf{X}, \mu_{\mathcal{I}}) \leq \epsilon(n)$ . We then use this algorithm to obtain the *multi-sample dynamic Gibbs sampling algorithm* that maintains  $N(n)$  independent samples for the current instance.

Given the error function  $\epsilon : \mathbb{N}^+ \rightarrow (0, 1)$ , suppose that  $T(\mathcal{I})$  is an easy-to-compute integer-valued function that upper bounds the mixing time on instance  $\mathcal{I}$ , such that

$$T(\mathcal{I}) \geq \tau_{\text{mix}}(\mathcal{I}, \epsilon(n)), \quad (6)$$

where  $\tau_{\text{mix}}(\mathcal{I}, \epsilon(n))$  denotes the mixing rate for the Gibbs sampling chain  $(\mathbf{X}_t)_{t \geq 0}$  on instance  $\mathcal{I}$ . By Proposition 8, if the Dobrushin-Shlosman condition is satisfied, we can set

$$T(\mathcal{I}) = \left\lceil \frac{n}{\delta} \log \frac{n}{\epsilon(n)} \right\rceil. \quad (7)$$

Our algorithm for single-sample dynamic Gibbs sampling maintains a random process  $(\mathbf{X}_t)_{t=0}^T$ , which is a Gibbs sampling chain on instance  $\mathcal{I}$  of length  $T = T(\mathcal{I})$ , where  $T(\mathcal{I})$  satisfies (6). Clearly  $\mathbf{X}_T$  is a sample for  $\mu_{\mathcal{I}}$  with  $d_{\text{TV}}(\mathbf{X}_T, \mu_{\mathcal{I}}) \leq \epsilon(n)$ .

When the current instance  $\mathcal{I}$  is updated to a new instance  $\mathcal{I}' = (V', E', Q, \Phi')$  where  $n' = |V'|$ , the original process  $(\mathbf{X}_t)_{t=0}^T$  is transformed to a new process  $(\mathbf{Y}_t)_{t=0}^{T'}$  such that the following holds as an invariant:  $(\mathbf{Y}_t)_{t=0}^{T'}$  is a Gibbs sampling chain on  $\mathcal{I}'$  with  $T' = T(\mathcal{I}')$ . Hence  $\mathbf{Y}_T$  is a sample for the new instance  $\mathcal{I}'$  with  $d_{\text{TV}}(\mathbf{Y}_T, \mu_{\mathcal{I}'}) \leq \epsilon(n')$ . This is achieved through the following two steps:

1. We construct couplings between  $(\mathbf{X}_t)_{t=0}^T$  and  $(\mathbf{Y}_t)_{t=0}^{T'}$ , so that the new process  $(\mathbf{Y}_t)_{t=0}^{T'}$  for  $\mathcal{I}'$  can be obtained by making small changes to the original process  $(\mathbf{X}_t)_{t=0}^T$  for  $\mathcal{I}$ .
2. We give a data structure which represents  $(\mathbf{X}_t)_{t=0}^T$  incrementally and supports various updates and queries to  $(\mathbf{X}_t)_{t=0}^T$  so that the above coupling can be generated efficiently.

The data structure is provided in the full version. In the following, we give the couplings.

## 6.1 Coupling for dynamic instances

The Gibbs sampling chain  $(\mathbf{X}_t)_{t=0}^T$  can be uniquely and fully recovered from: the initial state  $\mathbf{X}_0 \in Q^V$ , and the pairs  $\langle v_t, X_t(v_t) \rangle_{t=1}^T$  that record the transitions. We call  $\langle v_t, X_t(v_t) \rangle_{t=1}^T$  the *execution-log* for the chain  $(\mathbf{X}_t)_{t=0}^T$ , and denote it with

$$\text{Exe-Log}(\mathcal{I}, T) \triangleq \langle v_t, X_t(v_t) \rangle_{t=1}^T.$$



The following invariants are assumed for the random execution-log with an initial state.

► **Condition 10** (invariants for Exe-Log). *Fixed an initial state  $\mathbf{X}_0 \in Q^V$ , the followings hold for the random execution-log  $\text{Exe-Log}(\mathcal{I}, T) = \langle v_t, X_t(v_t) \rangle_{t=1}^T$  for the Gibbs sampling chain  $(\mathbf{X}_t)_{t=0}^T$  on instance  $\mathcal{I} = (V, E, Q, \Phi)$ :*

- $T = T(\mathcal{I})$  where  $T(\mathcal{I})$  satisfies (6);
- the random process  $(\mathbf{X}_t)_{t=0}^T$  uniquely recovered from the transitions  $\langle v_t, X_t(v_t) \rangle_{t=1}^T$  and the initial state  $\mathbf{X}_0$ , is identically distributed as the Gibbs sampling (Algorithm 1) on instance  $\mathcal{I}$  starting from initial state  $\mathbf{X}_0$  with  $v_t$  as the vertex picked at the  $t$ -th step.

Such invariants guarantee that  $\mathbf{X}_T$  provides a sample for  $\mu_{\mathcal{I}}$  with  $d_{TV}(\mathbf{X}_T, \mu_{\mathcal{I}}) \leq \epsilon(|V|)$ .

Suppose the current instance  $\mathcal{I}$  is updated to a new instance  $\mathcal{I}'$ . We construct couplings between the execution-log  $\text{Exe-Log}(\mathcal{I}, T) = \langle v_t, X_t(v_t) \rangle_{t=1}^T$  with initial state  $\mathbf{X}_0 \in Q^V$  for  $\mathcal{I}$  and the execution-log  $\text{Exe-Log}(\mathcal{I}', T') = \langle v'_t, Y_t(v'_t) \rangle_{t=1}^{T'}$  with initial state  $\mathbf{Y}_0 \in Q^{V'}$  for  $\mathcal{I}'$ . Our goal is as follows: assuming Condition 10 for  $\mathbf{X}_0$  and  $\text{Exe-Log}(\mathcal{I}, T)$ , the same condition should hold invariantly for  $\mathbf{Y}_0$  and  $\text{Exe-Log}(\mathcal{I}', T')$ .

Unlike traditional coupling of Markov chains for the analysis of mixing time, where the two chains start from arbitrarily distinct initial states but proceed by the same transition rule, here the two chains  $(\mathbf{X}_t)_{t=0}^T$  and  $(\mathbf{Y}_t)_{t=0}^{T'}$  start from similar states but have to obey different transition rules due to differences between instances  $\mathcal{I}$  and  $\mathcal{I}'$ .

Due to the technical reason, we divide the update from  $\mathcal{I} = (V, E, Q, \Phi)$  to  $\mathcal{I}' = (V', E', Q, \Phi')$  into two steps: we first update  $\mathcal{I} = (V, E, Q, \Phi)$  to

$$\mathcal{I}_{\text{mid}} = (V, E, Q, \Phi^{\text{mid}}), \quad (8)$$

where the potentials  $\Phi^{\text{mid}} = (\phi_a^{\text{mid}})_{a \in V \cup E}$  in the middle instance  $\mathcal{I}_{\text{mid}}$  are defined as

$$\forall a \in V \cup E, \quad \phi_a^{\text{mid}} \triangleq \begin{cases} \phi'_a & \text{if } a \in V' \cup E' \\ \phi_a & \text{if } a \notin V' \cup E'; \end{cases}$$

then we update  $\mathcal{I}_{\text{mid}} = (V, E, Q, \Phi^{\text{mid}})$  to  $\mathcal{I}' = (V', E', Q, \Phi')$ . In other words, the update from  $\mathcal{I}$  to  $\mathcal{I}_{\text{mid}}$  is only caused by updating the potentials of vertices and edges, while the underlying graph remains unchanged; and the update from  $\mathcal{I}_{\text{mid}}$  to  $\mathcal{I}'$  is only caused by updating the underlying graph, i.e. adding vertices, deleting vertices, adding edges and deleting edges.

The dynamic Gibbs sampling algorithm can be outlined as follows.

- **UpdateHamiltonian**: update  $\mathbf{X}_0$  and  $\langle v_t, X_t(v_t) \rangle_{t=1}^T$  to a new initial state  $\mathbf{Z}_0$  and a new execution log  $\text{Exe-Log}(\mathcal{I}_{\text{mid}}, T) = \langle u_t, Z_t(u_t) \rangle_{t=1}^T$  such that the random process  $(\mathbf{Z}_t)_{t=0}^T$  is the Gibbs sampling on instance  $\mathcal{I}_{\text{mid}}$ .
- **UpdateGraph**: update  $\mathbf{Z}_0$  and  $\langle u_t, Z_t(u_t) \rangle_{t=1}^T$  to a new initial state  $\mathbf{Y}_0$  and a new execution log  $\text{Exe-Log}(\mathcal{I}', T) = \langle v'_t, Y_t(v'_t) \rangle_{t=1}^{T'}$  such that the random process  $(\mathbf{Y}_t)_{t=0}^{T'}$  is the Gibbs sampling on instance  $\mathcal{I}'$ .
- **LengthFix**: change the length of the execution log  $\langle v'_t, Y_t(v'_t) \rangle_{t=1}^{T'}$  from  $T$  to  $T'$ , where  $T' = T(\mathcal{I}')$  and  $T(\mathcal{I}')$  satisfies (6).

The dynamic Gibbs sampling algorithm is given in Algorithm 2.

The subroutine **LengthFix** is given in Algorithm 3. The subroutine **UpdateGraph** is provided in the full version of the paper. In the following, we give the subroutines **UpdateHamiltonian**.

We consider the update of changing potentials of vertices and edges. The update do not change the underlying graph. Let  $\mathcal{I} = (V, E, Q, \Phi)$  be the current MRF instance. Let  $\mathbf{X}_0$  and  $\langle v_t, X_t(v_t) \rangle_{t=1}^T$  be the current initial state and execution log such that the random process  $(\mathbf{X}_t)_{t=0}^T$  is the Gibbs sampling on instance  $\mathcal{I}$ . Upon such an update, the new instance becomes



■ **Algorithm 2** Dynamic Gibbs sampling.

---

**Data** :  $\mathbf{X}_0 \in Q^V$  and  $\text{Exe-Log}(\mathcal{I}, T) = \langle v_t, X_t(v_t) \rangle_{t=1}^T$  for current  $\mathcal{I} = (V, E, Q, \Phi)$ .  
**Update**: an update that modifies  $\mathcal{I}$  to  $\mathcal{I}' = (V', E', Q, \Phi')$ .  
1 compute  $T' = T(\mathcal{I}')$  satisfying (6) and construct  $\mathcal{I}_{\text{mid}} = (V', E', Q, \Phi^{\text{mid}})$  as in (8);  
2  $(\mathbf{Z}_0, \langle u_t, Z_t(u_t) \rangle_{t=1}^T) \leftarrow \text{UpdateHamiltonian}(\mathcal{I}, \mathcal{I}_{\text{mid}}, \mathbf{X}_0, \langle v_t, X_t(v_t) \rangle_{t=1}^T)$ ;  
// update the potentials:  $\mathcal{I} \rightarrow \mathcal{I}_{\text{mid}}$   
3  $(\mathbf{Y}_0, \langle v'_t, Y_t(v'_t) \rangle_{t=1}^T) \leftarrow \text{UpdateGraph}(\mathcal{I}_{\text{mid}}, \mathcal{I}', \mathbf{Z}_0, \langle u_t, Z_t(u_t) \rangle_{t=1}^T)$ ;  
// update the underlying graph:  $\mathcal{I}_{\text{mid}} \rightarrow \mathcal{I}'$   
4  $(\mathbf{Y}_0, \langle v'_t, Y_t(v'_t) \rangle_{t=1}^{T'}) \leftarrow \text{LengthFix}(\mathcal{I}', \mathbf{Y}_0, \langle v'_t, Y_t(v'_t) \rangle_{t=1}^T, T')$ , where  $T' = T(\mathcal{I}')$ ;  
// change the length of the execution log from  $T$  to  $T' = T(\mathcal{I}')$   
5 update the data to  $\mathbf{Y}_0$  and  $\text{Exe-Log}(\mathcal{I}', T') = \langle v'_t, Y_t(v'_t) \rangle_{t=1}^{T'}$ ;

---

■ **Algorithm 3** LengthFix  $(\mathcal{I}, \mathbf{X}_0, \langle v_t, X_t(v_t) \rangle_{t=1}^T, T')$ .

---

**Data** :  $\mathbf{X}_0 \in Q^V$  and  $\text{Exe-Log}(\mathcal{I}, T) = \langle v_t, X_t(v_t) \rangle_{t=1}^T$  for current  $\mathcal{I} = (V, E, Q, \Phi)$ .  
**Input** : the new length  $T' > 0$ .  
1 **if**  $T' < T$  **then**  
2   truncate  $\langle v_t, X_t(v_t) \rangle_{t=1}^T$  to  $\langle v_t, X_t(v_t) \rangle_{t=1}^{T'}$ ;  
3 **else**  
4   extend  $\langle v_t, X_t(v_t) \rangle_{t=1}^T$  to  $\langle v_t, X_t(v_t) \rangle_{t=1}^{T'}$  by simulating the Gibbs sampling chain  
   on  $\mathcal{I}$  for  $T - T'$  more steps;  
5 update the data to  $\mathbf{X}_0$  and  $\text{Exe-Log}(\mathcal{I}, T') = \langle v_t, X_t(v_t) \rangle_{t=1}^{T'}$

---

$\mathcal{I}' = (V, E, Q, \Phi')$ . The algorithm  $\text{UpdateHamiltonian}(\mathcal{I}, \mathcal{I}', \mathbf{X}_0, \langle v_t, X_t(v_t) \rangle_{t=1}^T)$  updates the data to  $\mathbf{Y}_0$  and  $\langle v'_t, Y_t(v'_t) \rangle_{t=1}^T$  such that the random process  $(\mathbf{Y}_t)_{t=0}^T$  is the Gibbs sampling on instance  $\mathcal{I}'$ .

We transform the pair of  $\mathbf{X}_0 \in Q^V$  and  $\langle v_t, X_t(v_t) \rangle_{t=1}^T$  to a new pair of  $\mathbf{Y}_0 \in Q^V$  and  $\langle v_t, Y_t(v_t) \rangle_{t=1}^T$  for  $\mathcal{I}'$ . This is achieved as follows: the vertex sequence  $(v_t)_{t=1}^T$  is identically coupled and the chain  $(\mathbf{X}_t)_{t=0}^T$  is transformed to  $(\mathbf{Y}_t)_{t=0}^T$  by the following one-step local coupling between  $\mathbf{X}$  and  $\mathbf{Y}$ .

► **Definition 11** (one-step local coupling for Hamiltonian update). *The two chains  $(\mathbf{X}_t)_{t=0}^\infty$  on instance  $\mathcal{I} = (V, E, Q, \Phi)$  and  $(\mathbf{Y}_t)_{t=0}^\infty$  on instance  $\mathcal{I}' = (V, E, Q, \Phi')$  are coupled as:*

- Initially  $\mathbf{X}_0 = \mathbf{Y}_0 \in Q^V$ ;
- for  $t = 1, 2, \dots$ , the two chains  $\mathbf{X}$  and  $\mathbf{Y}$  jointly do:
  1. pick the same  $v_t \in V$ , and let  $(X_t(u), Y_t(u)) \leftarrow (X_{t-1}(u), Y_{t-1}(u))$  for all  $u \in V \setminus \{v_t\}$ ;
  2. sample  $(X_t(v_t), Y_t(v_t))$  from a coupling  $D_{\mathcal{I}_v, \mathcal{I}'_v}^{\sigma, \tau}(\cdot, \cdot)$  of the marginal distributions  $\mu_{v_t, \mathcal{I}}(\cdot \mid \sigma)$  and  $\mu_{v_t, \mathcal{I}'}(\cdot \mid \tau)$  with  $\sigma = X_{t-1}(\Gamma_G(v_t))$  and  $\tau = Y_{t-1}(\Gamma_G(v_t))$ , where  $G = (V, E)$ .

The local coupling  $D_{\mathcal{I}_v, \mathcal{I}'_v}^{\sigma, \tau}(\cdot, \cdot)$  for Hamiltonian update is specified as follows.

► **Definition 12** (local coupling  $D_{\mathcal{I}_v, \mathcal{I}'_v}^{\sigma, \tau}(\cdot, \cdot)$  for Hamiltonian update). *Let  $v \in V$  be vertex and  $\sigma, \tau \in Q^{\Gamma_G(v)}$  two configurations, where  $G = (V, E)$ . We say a random pair  $(c, c') \in Q^2$  is drawn from the coupling  $D_{\mathcal{I}_v, \mathcal{I}'_v}^{\sigma, \tau}(\cdot, \cdot)$  if  $(c, c')$  is generated by the following two steps:*

- **sampling step:** sample  $(c, c') \in Q^2$  jointly from an optimal coupling  $D_{\text{opt}, \mathcal{I}_v}^{\sigma, \tau}$  of the marginal distributions  $\mu_{v, \mathcal{I}}(\cdot | \sigma)$  and  $\mu_{v, \mathcal{I}}(\cdot | \tau)$ , such that  $c \sim \mu_{v, \mathcal{I}}(\cdot | \sigma)$  and  $c' \sim \mu_{v, \mathcal{I}}(\cdot | \tau)$ ;
- **resampling step:** flip a coin independently with the probability of HEADS being

$$p_{\mathcal{I}_v, \mathcal{I}'_v}^{\tau}(c') \triangleq \begin{cases} 0 & \text{if } \mu_{v, \mathcal{I}}(c' | \tau) \leq \mu_{v, \mathcal{I}'}(c' | \tau), \\ \frac{\mu_{v, \mathcal{I}}(c' | \tau) - \mu_{v, \mathcal{I}'}(c' | \tau)}{\mu_{v, \mathcal{I}}(c' | \tau)} & \text{otherwise;} \end{cases} \quad (9)$$

if the outcome of coin flipping is HEADS, resample  $c'$  from the distribution  $\nu_{\mathcal{I}_v, \mathcal{I}'_v}^{\tau}$  independently, where the distribution  $\nu_{\mathcal{I}_v, \mathcal{I}'_v}^{\tau}$  is defined as

$$\forall b \in Q: \quad \nu_{\mathcal{I}_v, \mathcal{I}'_v}^{\tau}(b) \triangleq \frac{\max\{0, \mu_{v, \mathcal{I}'}(b | \tau) - \mu_{v, \mathcal{I}}(b | \tau)\}}{\sum_{x \in Q} \max\{0, \mu_{v, \mathcal{I}}(x | \tau) - \mu_{v, \mathcal{I}'}(x | \tau)\}}. \quad (10)$$

► **Lemma 13.**  $D_{\mathcal{I}_v, \mathcal{I}'_v}^{\sigma, \tau}(\cdot, \cdot)$  in Definition 12 is a valid coupling between  $\mu_{v, \mathcal{I}}(\cdot | \sigma)$  and  $\mu_{v, \mathcal{I}'}(\cdot | \tau)$ .

By Lemma 13, the resulting  $(\mathbf{Y}_t)_{t=0}^T$  is a faithful copy of the Gibbs sampling on instance  $\mathcal{I}'$ , assuming that  $(\mathbf{X}_t)_{t=0}^T$  is such a chain on instance  $\mathcal{I}$ .

Next we give an upper bound for the probability  $p_{\mathcal{I}_v, \mathcal{I}'_v}^{\tau}(\cdot)$  defined in (9).

► **Lemma 14.** For any two instances  $\mathcal{I} = (V, E, Q, \Phi)$  and  $\mathcal{I}' = (V, E, Q, \Phi')$  of MRF model, and any  $v \in V, c \in Q$  and  $\sigma \in Q^{\Gamma_G(v)}$ , it holds that

$$p_{\mathcal{I}_v, \mathcal{I}'_v}^{\tau}(c) \leq 2 \left( \|\phi_v - \phi'_v\|_1 + \sum_{e=\{u, v\} \in E} \|\phi_e - \phi'_e\|_1 \right), \quad (11)$$

where  $\|\phi_v - \phi'_v\|_1 = \sum_{c \in Q} |\phi_v(c) - \phi'_v(c)|$  and  $\|\phi_e - \phi'_e\|_1 = \sum_{c, c' \in Q} |\phi_e(c, c') - \phi'_e(c, c')|$ .

By Lemma 14, for each vertex  $v \in V$ , we define an upper bound of the probability  $p_{\mathcal{I}_v, \mathcal{I}'_v}^{\tau}(\cdot)$  as

$$p_v^{\text{up}} \triangleq \min \left\{ 2 \left( \|\phi_v - \phi'_v\|_1 + \sum_{e=\{u, v\} \in E} \|\phi_e - \phi'_e\|_1 \right), 1 \right\}. \quad (12)$$

With  $p_v^{\text{up}}$ , we can implement the one-step local coupling in Definition 11 as follows. We first sample each  $v_i \in V$  for  $1 \leq i \leq T$  uniformly and independently. For each vertex  $v \in V$ , let  $T_v \triangleq \{1 \leq t \leq T \mid v_t = v\}$  be the set of all the steps that pick the vertex  $v$ . We select each  $t \in T_v$  independently with probability  $p_v^{\text{up}}$  to construct a random subset  $\mathcal{P}_v \subseteq T_v$ , and let  $\mathcal{P} \triangleq \bigcup_{v \in V} \mathcal{P}_v$ . We then couple the two chains  $(\mathbf{X}_t)_{t=0}^T$  and  $(\mathbf{Y}_t)_{t=0}^T$ . First set  $\mathbf{X}_0 = \mathbf{Y}_0$ . For each  $1 \leq t \leq T$ , we set  $(X_t(u), Y_t(u)) \leftarrow (X_{t-1}(u), Y_{t-1}(u))$  for all  $u \in V \setminus \{v_t\}$ ; then generate the random pair  $(X_t(v_t), Y_t(v_t))$  by the following procedure.

- **sampling step:** Let  $\sigma = X_{t-1}(\Gamma_G(v_t))$  and  $\tau = Y_{t-1}(\Gamma_G(v_t))$ . We draw a random pair  $(c, c') \in Q^2$  from the optimal coupling  $D_{\text{opt}, \mathcal{I}_v}^{\sigma, \tau}$  of the marginal distributions  $\mu_{v, \mathcal{I}}(\cdot | \sigma)$  and  $\mu_{v, \mathcal{I}}(\cdot | \tau)$  such that  $c \sim \mu_{v, \mathcal{I}}(\cdot | \sigma)$  and  $c' \sim \mu_{v, \mathcal{I}}(\cdot | \tau)$ ;
- **resampling step:** If  $t \notin \mathcal{P}$ , set  $X_t(v_t) = c$  and  $Y_t(v_t) = c'$ . Otherwise, set  $X_t(v_t) = c$  and

$$Y_t(v_t) = \begin{cases} b \sim \nu_{\mathcal{I}_{v_t}, \mathcal{I}'_{v_t}}^{\tau} & \text{with probability } p_{\mathcal{I}_{v_t}, \mathcal{I}'_{v_t}}^{\tau}(c')/p_{v_t}^{\text{up}} \\ c' & \text{with probability } 1 - p_{\mathcal{I}_{v_t}, \mathcal{I}'_{v_t}}^{\tau}(c')/p_{v_t}^{\text{up}}. \end{cases} \quad (13)$$

Note that  $p_{v_t}^{\text{up}} > 0$  if  $t \in \mathcal{P}$ . By Lemma 14, it must hold that  $p_{\mathcal{I}_{v_t}, \mathcal{I}'_{v_t}}^{\tau}(c') \leq p_{v_t}^{\text{up}}$ . Hence, the probability  $p_{\mathcal{I}_{v_t}, \mathcal{I}'_{v_t}}^{\tau}(c')/p_{v_t}^{\text{up}}$  is valid. Note that the probability that  $Y_t(v_t)$  is set as  $b$  is

$$\Pr[Y_t(v_t) \text{ is set as } b] = \Pr[t \in \mathcal{P}] \cdot \frac{p_{\mathcal{I}_{v_t}, \mathcal{I}'_{v_t}}^{\tau}(c')}{p_{v_t}^{\text{up}}} = p_{v_t}^{\text{up}} \cdot \frac{p_{\mathcal{I}_{v_t}, \mathcal{I}'_{v_t}}^{\tau}(c')}{p_{v_t}^{\text{up}}} = p_{\mathcal{I}_{v_t}, \mathcal{I}'_{v_t}}^{\tau}(c').$$

Hence, our implementation perfectly simulates the coupling in Definition 11.

Let  $\mathcal{D}_t$  denote the *set of disagreements* between  $\mathbf{X}_t$  and  $\mathbf{Y}_t$ . Formally,

$$\mathcal{D}_t \triangleq \{v \in V \mid X_t(v) \neq Y_t(v)\}.$$

Note that if  $v_t \notin \Gamma_G(\mathcal{D}_{t-1})$ , the random pair  $(c, c')$  drawn from the coupling  $D_{\text{opt}, \mathcal{I}_v}^{\sigma, \tau}$  must satisfy  $c = c'$ . Thus it is easy to make the following observation for the  $(\mathbf{X}_t)_{t=0}^T$  and  $(\mathbf{Y}_t)_{t=0}^T$  coupled as above.

► **Observation 15.** *For any integer  $t \in [1, T]$ , if  $v_t \notin \Gamma_G^+(\mathcal{D}_{t-1})$  and  $t \notin \mathcal{P}$ , then  $X_t(v_t) = Y_t(v_t)$  and  $\mathcal{D}_t = \mathcal{D}_{t-1}$ .*

With this observation, the new  $\mathbf{Y}_0$  and  $\text{Exe-Log}(\mathcal{I}', T) = \langle v_t, Y_t(v_t) \rangle_{t=1}^T$  can be generated from  $\mathbf{X}_0$  and  $\text{Exe-Log}(\mathcal{I}, T) = \langle v_t, X_t(v_t) \rangle_{t=1}^T$  as Algorithm 4.

Observation 15 says that the nontrivial coupling between  $X_t(v_t)$  and  $Y_t(v_t)$  is only needed when  $v_t \in \Gamma_G^+(\mathcal{D}_{t-1})$  or  $t \in \mathcal{P}$ , which occurs rarely as long as  $\mathcal{D}_{t-1}$  and  $\mathcal{P}$  are small. This is a key to ensure the small incremental time cost of Algorithm 4. The following lemma bounds the expected cost for UpdateHamiltonian.

► **Lemma 16** (cost of the coupling for UpdateHamiltonian). *Let  $\mathcal{I} = (V, E, Q, \Phi)$  be the current MRF instance and  $\mathcal{I}' = (V, E, Q, \Phi')$  the updated instance. Assume that  $\mathcal{I}$  satisfies Dobrushin-Shlosman condition (Condition 4) with constant  $\delta > 0$ , and  $d_{\text{Hamil}}(\mathcal{I}, \mathcal{I}') = \sum_{v \in V} \|\phi_v - \phi'_v\|_1 + \sum_{e \in E} \|\phi_e - \phi'_e\|_1 \leq L$ . It holds that  $\mathbb{E} \left[ \sum_{t=1}^T \mathbf{1}[t \in \mathcal{P} \vee v_t \in \Gamma_G^+(\mathcal{D}_{t-1})] \right] = O\left(\frac{\Delta T L}{n\delta}\right)$ , where  $n = |V|$ ,  $\Delta$  is the maximum degree of graph  $G = (V, E)$ .*

## 6.2 Dynamic Gibbs sampling algorithm

The couplings constructed in Section 6.1 can be implemented as the algorithm for dynamic Gibbs sampling. Recall  $d_{\text{graph}}(\cdot, \cdot)$  and  $d_{\text{Hamil}}(\cdot, \cdot)$  are defined in (2).

► **Lemma 17** (single-sample dynamic Gibbs sampling algorithm). *Let  $\epsilon : \mathbb{N}^+ \rightarrow (0, 1)$  be an error function. Let  $\mathcal{I} = (V, E, Q, \Phi)$  be an MRF instance with  $n = |V|$  and  $\mathcal{I}' = (V', E', Q, \Phi')$  the updated instance with  $n' = |V'|$ . Denote  $T = T(\mathcal{I})$ ,  $T' = T(\mathcal{I}')$  and  $T_{\max} = \max\{T, T'\}$ . Assume  $d_{\text{graph}}(\mathcal{I}, \mathcal{I}') \leq L_{\text{graph}} = o(n)$ ,  $d_{\text{Hamil}}(\mathcal{I}, \mathcal{I}') \leq L_{\text{Hamil}}$ , and  $T, T' \in \Omega(n \log n)$ . The single-sample dynamic Gibbs sampling algorithm (Algorithm 2) does the followings:*

- **(space cost)** *The algorithm maintains an explicit copy of a sample  $\mathbf{X} \in Q^V$  for the current instance  $\mathcal{I}$ , and also a data structure using  $O(T)$  memory words, each of  $O(\log T)$  bits, for representing an initial state  $\mathbf{X}_0 \in Q^V$  and an execution-log  $\text{Exe-Log}(\mathcal{I}, T) = \langle v_t, X_t(v_t) \rangle_{t=1}^T$  for the Gibbs sampling  $(\mathbf{X}_t)_{t=0}^T$  on  $\mathcal{I}$  generating sample  $\mathbf{X} = \mathbf{X}_T$ .*
- **(correctness)** *Assuming that Condition 10 holds for  $\mathbf{X}_0$  and  $\text{Exe-Log}(\mathcal{I}, T)$  for the Gibbs sampling on  $\mathcal{I}$ , upon each update that modifies  $\mathcal{I}$  to  $\mathcal{I}'$ , the algorithm updates  $\mathbf{X}$  to an explicit copy of a sample  $\mathbf{Y} \in Q^{V'}$  for the new instance  $\mathcal{I}'$ , and correspondingly updates the  $\mathbf{X}_0$  and  $\text{Exe-Log}(\mathcal{I}, T)$  represented by the data structure to a  $\mathbf{Y}_0 \in Q^{V'}$  and  $\text{Exe-Log}(\mathcal{I}', T') = \langle v'_t, Y_t(v'_t) \rangle_{t=1}^{T'}$  for the Gibbs sampling  $(\mathbf{Y}_t)_{t=0}^{T'}$  on  $\mathcal{I}'$  generating the new sample  $\mathbf{Y} = \mathbf{Y}_{T'}$ , where  $\mathbf{Y}_0$  and  $\text{Exe-Log}(\mathcal{I}', T')$  satisfy Condition 10 for the Gibbs sampling on  $\mathcal{I}'$ , therefore,*

$$d_{\text{TV}}(\mathbf{Y}, \mu_{\mathcal{I}'}^*) \leq \epsilon(n').$$

■ **Algorithm 4** UpdateHamiltonian  $(\mathcal{I}, \mathcal{I}', \mathbf{X}_0, \langle v_t, X_t(v_t) \rangle_{t=1}^T)$ .

---

**Data** :  $\mathbf{X}_0 \in Q^V$  and  $\text{Exe-Log}(\mathcal{I}, T) = \langle v_t, X_t(v_t) \rangle_{t=1}^T$  for  $\mathcal{I} = (V, E, Q, \Phi)$ .  
**Update**: an update that modifies  $\mathcal{I}$  to  $\mathcal{I}' = (V, E, Q, \Phi')$ .

- 1  $t_0 \leftarrow 0$ ,  $\mathcal{D} \leftarrow \emptyset$ , and construct a  $\mathbf{Y}_0 \leftarrow \mathbf{X}_0$ ;
- 2 for each  $v \in V$ , construct a random subset  $\mathcal{P}_v \subseteq T_v \triangleq \{1 \leq t \leq T \mid v_t = v\}$  such that each element in  $T_v$  is selected independently with probability  $p_v^{\text{up}}$  defined in (12);
- 3 construct the set  $\mathcal{P} \leftarrow \bigcup_{v \in V} \mathcal{P}_v$ ;
- 4 **while**  $\exists t_0 < t \leq T$  such that  $v_t \in \Gamma_G^+(\mathcal{D})$  or  $t \in \mathcal{P}$  **do**
- 5     find the smallest  $t > t_0$  such that  $v_t \in \Gamma_G^+(\mathcal{D})$  or  $t \in \mathcal{P}$ ;
- 6     for all  $t_0 < i < t$ , let  $Y_i(v_i) = X_i(v_i)$ ;
- 7     sample  $Y_t(v_t) \in Q$  conditioning on  $X_t(v_t)$  according to the optimal coupling between  $\mu_{v_t, \mathcal{I}}(\cdot \mid X_{t-1}(\Gamma_G(v_t)))$  and  $\mu_{v_t, \mathcal{I}'}(\cdot \mid Y_{t-1}(\Gamma_G(v_t)))$ ;
- 8     **if**  $t \in \mathcal{P}$  **then**
- 9         **with probability**  $p_{\mathcal{I}_{v_t}, \mathcal{I}'_{v_t}}^\tau(Y_t(v_t))/p_{v_t}^{\text{up}}$  where  $\tau = Y_{t-1}(\Gamma_G(v_t))$  **do**
- 10             resample  $Y_t(v_t) \sim \nu_{\mathcal{I}_{v_t}, \mathcal{I}'_{v_t}}^\tau$ , where  $\nu_{\mathcal{I}_{v_t}, \mathcal{I}'_{v_t}}^\tau$  is defined in (10) ;
- 11     **if**  $X_t(v_t) \neq Y_t(v_t)$  **then**  $\mathcal{D} \leftarrow \mathcal{D} \cup \{v_t\}$  **else**  $\mathcal{D} \leftarrow \mathcal{D} \setminus \{v_t\}$ ;
- 12      $t_0 \leftarrow t$ ;
- 13 for all remaining  $t_0 < i \leq T$ : let  $Y_i(v_i) = X_i(v_i)$ ;
- 14 update the data to  $\mathbf{Y}_0$  and  $\text{Exe-Log}(\mathcal{I}', T) = \langle v_t, Y_t(v_t) \rangle_{t=1}^T$ ;

---

■ **(time cost)** Assuming Condition 10 for  $\mathbf{X}_0$  and  $\text{Exe-Log}(\mathcal{I}, T)$  for the Gibbs sampling on  $\mathcal{I}$ , the expected time complexity for resolving an update is

$$O\left(\Delta n + \Delta\left(|T - T'| + \left(\Delta \log n + \frac{T_{\max}}{n}\right)(L_{\text{Hamil}} + L_{\text{graph}})\right) \log^2 T_{\max}\right),$$

where  $\Delta = \max\{\Delta_G, \Delta_{G'}\}$ ,  $\Delta_G, \Delta_{G'}$  denote the maximum degrees of  $G = (V, E)$  and  $G' = (V', E')$ .

We remark that the  $O(\Delta n)$  in time cost is necessary because the update from  $\mathcal{I}$  to  $\mathcal{I}'$  may change all the potentials of vertices and edges. One can reduce the  $O(\Delta n)$  from the time cost if we further restrict that one update can only change constant number of vertices, edges, and potentials.

One can extend Algorithm 2 to an *Multi-sample dynamic Gibbs sampling algorithm* that maintains multiple independent random samples for the current MRF instance. By Lemma 17, it is easy to prove that the Multi-sample algorithm is correct and efficient. Thus Theorem 9 follows immediately. The detail of the Multi-sample dynamic Gibbs sampling algorithm and the proof of Theorem 9 are provided in the full version of the paper.

## 7 Conclusion

In this paper we study probabilistic inference problem in a graphical model when the model itself is changing dynamically with time. We study the non-local updates so that two consecutive graphical models may differ everywhere as long as the total amount of their difference is bounded. This general setting covers many typical applications. We give a sampling-based dynamic inference algorithm that maintains an inference solution efficiently against the dynamic inputs. The algorithm significantly improves the time cost compared to the static sampling-based inference algorithm.

Our algorithm generically reduces the dynamic inference to dynamic sampling problem. Our main technical contribution is a dynamic Gibbs sampling algorithm that maintains random samples for graphical models dynamically changed by non-local updates. Such technique is extendable to all single-site dynamics. This gives us a systematic approach for transforming classic MCMC samplers on static inputs to the sampling and inference algorithms in a dynamic setting. Our dynamic algorithms are efficient as long as the one-step optimal coupling exhibits a step-wise decay, a key property that has been widely used in supporting efficient MCMC sampling in the classic static setting and captured by the Dobrushin-Shlosman condition.

Our result is the first one that shows the possibility of efficient probabilistic inference in dynamically changing graphical models (especially when the graphical models are changed by non-local updates). Our dynamic inference algorithm has potentials in speeding up the iterative algorithms for learning graphical models, which deserves more theoretical and experimental research. In this paper, we focus on discrete graphical models and sampling-based inference algorithms. Important future directions include considering more general distributions and the dynamic algorithms based on other inference techniques.

---

## References

- 1 Ittai Abraham, David Durfee, Ioannis Koutis, Sebastian Krinninger, and Richard Peng. On fully dynamic graph sparsifiers. In *FOCS*, 2016.
- 2 Osvaldo Anacleto, Catriona Queen, et al. Dynamic chain graph models for time series network data. *Bayesian Anal.*, 12(2):491–509, 2017.
- 3 Aaron Bernstein and Shiri Chechik. Deterministic decremental single source shortest paths: beyond the  $o(mn)$  bound. In *STOC*, 2016.
- 4 Russ Bubley and Martin Dyer. Path coupling: A technique for proving rapid mixing in Markov chains. In *FOCS*, 1997.
- 5 Carlos M. Carvalho and Mike West. Dynamic matrix-variate graphical models. *Bayesian Anal.*, 2(1):69–97, 2007.
- 6 Christopher De Sa, Kunle Olukotun, and Christopher Ré. Ensuring rapid mixing and low bias for asynchronous Gibbs sampling. In *ICML*, 2016.
- 7 RL Dobrushin and SB Shlosman. Completely analytical interactions: constructive description. *J. Statist. Phys.*, 46(5-6):983–1014, 1987.
- 8 Roland L Dobrushin and Senya B Shlosman. Completely analytical Gibbs fields. In *Statistical Physics and Dynamical Systems*, pages 371–403. Springer, 1985.
- 9 Roland Lvovich Dobrushin and Senya B Shlosman. Constructive criterion for the uniqueness of Gibbs field. In *Statistical Physics and Dynamical Systems*, pages 347–370. Springer, 1985.
- 10 David Durfee, Yu Gao, Gramoz Goranci, and Richard Peng. Fully dynamic effective resistances. *arXiv preprint*, 2018. [arXiv:1804.04038](#).
- 11 David Durfee, Yu Gao, Gramoz Goranci, and Richard Peng. Fully dynamic spectral vertex sparsifiers and applications. In *STOC*, 2019.
- 12 Martin Dyer, Leslie Ann Goldberg, and Mark Jerrum. Dobrushin conditions and systematic scan. *Combin. Probab. Comput.*, 17(6):761–779, 2008.
- 13 Martin Dyer and Catherine Greenhill. On Markov chains for independent sets. *J. Algorithms*, 35(1):17–49, 2000.
- 14 Weiming Feng, Nisheeth K Vishnoi, and Yitong Yin. Dynamic sampling from graphical models. In *STOC*, 2019.
- 15 Sebastian Forster and Gramoz Goranci. Dynamic low-stretch trees via dynamic low-diameter decompositions. In *STOC*, pages 377–388, 2019.
- 16 Andreas Galanis, Daniel Štefankovič, and Eric Vigoda. Inapproximability for antiferromagnetic spin systems in the tree nonuniqueness region. *J. ACM*, 62(6):50, 2015.

- 17 Andreas Galanis, Daniel Štefankovič, and Eric Vigoda. Inapproximability of the partition function for the antiferromagnetic Ising and hard-core models. *Combin. Probab. Comput.*, 25(04):500–559, 2016.
- 18 Gramoz Goranci, Monika Henzinger, and Pan Peng. Dynamic Effective Resistances and Approximate Schur Complement on Separable Graphs. In *ESA*, volume 112, 2018.
- 19 Thomas P Hayes. A simple condition implying rapid mixing of single-site dynamics on spin systems. In *FOCS*, 2006.
- 20 Monika Henzinger, Sebastian Krinninger, and Danupon Nanongkai. Decremental single-source shortest paths on undirected graphs in near-linear total update time. In *FOCS*, 2014.
- 21 Monika Henzinger, Sebastian Krinninger, and Danupon Nanongkai. Dynamic approximate all-pairs shortest paths: Breaking the  $O(mn)$  barrier and derandomization. *SIAM J. Comput.*, 45(3):947–1006, 2016.
- 22 Geoffrey E Hinton. A practical guide to training restricted boltzmann machines. In *Neural Networks: Tricks of the Trade*, pages 599–619. Springer, 2012.
- 23 Mark Jerrum. A very simple algorithm for estimating the number of  $k$ -colorings of a low-degree graph. *Random Structures & Algorithms*, 7(2):157–165, 1995.
- 24 Mark Jerrum, Leslie G. Valiant, and Vijay V. Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theoret. Comput. Sci.*, 43:169–188, 1986.
- 25 Daphne Koller, Nir Friedman, and Francis Bach. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.
- 26 Holden Lee, Oren Mangoubi, and Nisheeth Vishnoi. Online sampling from log-concave distributions. In *NIPS*, 2019.
- 27 David A Levin and Yuval Peres. *Markov chains and mixing times*. American Mathematical Soc., 2017.
- 28 Michael Luby and Eric Vigoda. Fast convergence of the Glauber dynamics for sampling independent sets. *Random Structures & Algorithms*, 15(3-4):229–241, 1999.
- 29 Marc Mezard and Andrea Montanari. *Information, physics, and computation*. Oxford University Press, 2009.
- 30 Danupon Nanongkai, Thatchaphol Saranurak, and Christian Wulff-Nilsen. Dynamic minimum spanning forest with subpolynomial worst-case update time. In *FOCS*, 2017.
- 31 Hariharan Narayanan and Alexander Rakhlin. Efficient sampling from time-varying log-concave distributions. *J. Mach. Learn. Res.*, 18(1):4017–4045, 2017.
- 32 Catriona M. Queen and Jim Q. Smith. Multiregression dynamic models. *J. Roy. Statist. Soc. Ser. B*, 55(4):849–870, 1993.
- 33 Cedric Renggli, Bojan Karlaš, Bolin Ding, Feng Liu, Kevin Schawinski, Wentao Wu, and Ce Zhang. Continuous integration of machine learning models: A rigorous yet practical treatment. In *SysML*, 2019.
- 34 Padhraic Smyth, Max Welling, and Arthur U Asuncion. Asynchronous distributed learning of topic models. In *NIPS*, 2009.
- 35 Daniel Štefankovič, Santosh Vempala, and Eric Vigoda. Adaptive simulated annealing: A near-optimal connection between sampling and counting. *J. ACM*, 56(3):18, 2009.
- 36 Eric Vigoda. Fast convergence of the Glauber dynamics for sampling independent sets: Part II. Technical Report TR-99-003, International Computer Science Institute, 1999.
- 37 Martin J. Wainwright and Michael I. Jordan. *Graphical models, exponential families, and variational inference*. Now Publishers Inc, 2008.
- 38 Christian Wulff-Nilsen. Fully-dynamic minimum spanning forest with improved worst-case update time. In *STOC*, 2017.

# Theorems of KKL, Friedgut, and Talagrand via Random Restrictions and Log-Sobolev Inequality

**Esty Kelman**

School of Computer Science, Tel Aviv University, Israel  
estykelman@gmail.com

**Subhash Khot**

Department of Computer Science, Courant Institute of Mathematical Sciences, New York University, NY, USA  
khot@cims.nyu.edu

**Guy Kindler**

Einstein Institute of Mathematics, Hebrew University of Jerusalem, Israel  
gkindler@cs.huji.ac.il

**Dor Minzer**

Department of Mathematics, Massachusetts Institute of Technology, Cambridge, MA, USA  
minzer.dor@gmail.com

**Muli Safra**

School of Computer Science, Tel Aviv University, Israel  
muli.safra@gmail.com

---

## Abstract

We give alternate proofs for three related results in analysis of Boolean functions, namely the KKL Theorem, Friedgut's Junta Theorem, and Talagrand's strengthening of the KKL Theorem. We follow a new approach: looking at the first Fourier level of the function after a suitable random restriction and applying the Log-Sobolev inequality appropriately. In particular, we avoid using the hypercontractive inequality that is common to the original proofs. Our proofs might serve as an alternate, uniform exposition to these theorems and the techniques might benefit further research.

**2012 ACM Subject Classification** Mathematics of computing → Discrete mathematics

**Keywords and phrases** Fourier Analysis, Hypercontractivity, Log-Sobolev Inequality

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.26

**Funding** *Esty Kelman*: Supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (Grant agreement No. 835152).

*Subhash Khot*: Supported by the NSF Award CCF-1422159, the Simons Collaboration on Algorithms and Geometry, and the Simons Investigator Award.

*Muli Safra*: Supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (Grant agreement No. 835152).

## 1 Introduction

Let us consider the Boolean cube  $\{0,1\}^n$  equipped with the uniform measure and let  $f: \{0,1\}^n \rightarrow \{0,1\}$  be a function. The influence of a coordinate  $i \in [n]$ , denoted by  $I_i[f]$ , is defined to be  $\Pr_x[f(x) \neq f(x \oplus e_i)]$ , where  $x \in \{0,1\}^n$  is sampled uniformly and  $x \oplus e_i$  denotes the input  $x$  with the  $i^{\text{th}}$  bit flipped. The total influence of  $f$  is  $I[f] = \sum_{i=1}^n I_i[f]$ . One of the most basic inequalities, known as Poincare's inequality, states that  $I[f] \geq \text{var}(f)$ , where  $\text{var}(f)$  is the variance of the random variable  $f(x)$  when  $x \in \{0,1\}^n$  is sampled uniformly. In general, Poincare's inequality may be tight, which raises the following question: can it



© Esty Kelman, Subhash Khot, Guy Kindler, Dor Minzer, and Muli Safra;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 26; pp. 26:1–26:17



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



be the case that not only  $I[f] \approx \text{var}(f)$ , but actually  $I_i[f] \approx \frac{\text{var}(f)}{n}$  for all  $i \in [n]$ ? In other words, can all influences of  $f$  be as small as possible simultaneously? The landmark result of Kahn, Kalai, and Linial [13] gives a negative answer to this question:

► **Theorem 1.** *There exists an absolute constant  $c > 0$ , such that for any  $f: \{0, 1\}^n \rightarrow \{0, 1\}$ , there is a coordinate  $i \in [n]$  with  $I_i[f] \geq c \cdot \frac{\log n}{n} \text{var}(f)$ .*

The KKL Theorem and its strengthenings by Friedgut [9] and Talagrand [19] are foundational results in analysis of Boolean functions. These have found several applications, e.g. to the threshold phenomena, computational learning theory, extremal combinatorics, communication complexity, hardness of approximation, non-embeddability results in metric geometry, and coding theory [10, 18, 5, 11, 3, 6, 14, 15, 4, 16]. Before we discuss the theorems of Friedgut and Talagrand, let us state a dimension-free variant of the KKL Theorem (that is morally equivalent to Theorem 1 and is easily implied by the techniques in [13]).

► **Theorem 2.** *There exists an absolute constant  $K > 0$ , such that for any  $f: \{0, 1\}^n \rightarrow \{0, 1\}$ , there is a coordinate  $i \in [n]$  with  $I_i[f] \geq 2^{-K \frac{I[f]}{\text{var}(f)}}$ .*

We note that Theorem 2 implies Theorem 1: if  $I[f] \geq \frac{\log n}{2^K} \text{var}(f)$ , then clearly there is a coordinate  $i \in [n]$  such that  $I_i[f] \geq \frac{I[f]}{n} \geq \frac{1}{2^K} \frac{\log n}{n} \text{var}(f)$ . Otherwise, by Theorem 2, there is a coordinate  $i \in [n]$  such that  $I_i[f] \geq 2^{-K \frac{I[f]}{\text{var}(f)}} \geq \frac{1}{\sqrt{n}}$  and we are done either way. Friedgut's Junta Theorem can now be stated as below.

► **Theorem 3.** *There exists an absolute constant  $K > 0$ , such that for any  $f: \{0, 1\}^n \rightarrow \{0, 1\}$  and  $\varepsilon > 0$ , the function  $f$  is  $\varepsilon$ -close to a function  $g: \{0, 1\}^n \rightarrow \{0, 1\}$  (in Hamming distance) that depends on at most  $2^{K \frac{I[f]}{\varepsilon}}$  coordinates.*

Morally speaking, Theorem 3 states that not only that there is a coordinate with significant influence as in Theorem 2, but actually all coordinates that have smaller influence, combined, barely affect the output of the function  $f$  (and this is how its proof proceeds). Talagrand's strengthening of the KKL Theorem is stated below.

► **Theorem 4.** *There exists an absolute constant  $c > 0$ , such that for any  $f: \{0, 1\}^n \rightarrow \{0, 1\}$ ,*

$$\sum_{i=1}^n \frac{I_i[f]}{\log(1/I_i[f])} \geq c \cdot \text{var}(f).$$

We note that Theorem 4 implies Theorem 2 as follows: suppose on the contrary that all influences  $I_i[f]$  are at most  $2^{-K \frac{I[f]}{\text{var}(f)}}$ . Then the “Talagrand sum” as above is at most  $\frac{\text{var}(f)}{K I[f]} \sum_{i=1}^n I_i[f] = \frac{\text{var}(f)}{K}$ , a contradiction for a large enough constant  $K$ .

A key technique used in the original proofs of all the theorems above is the hypercontractive inequality (stated in Section 2.3). The use of this inequality is, by now, nearly ubiquitous in analysis of Boolean functions. Still, using this inequality might impose limitations of its own, limiting the discovery of new results, both qualitatively and quantitatively. As far as we know, researchers in this area have wondered whether there is “life” beyond the hypercontractive inequality, and certainly there have been efforts to prove the KKL Theorem (and its strengthenings) without using it. In particular, proofs using “only” the Log-Sobolev inequality (stated in Section 2.3) for the KKL Theorem and Friedgut's Junta Theorem are known [8] (their argument though does not seem to extend to Talagrand's Theorem). There is also a recent proof of the KKL Theorem (as well as Talagrand's result and some strengthenings) using stochastic calculus [7].

In this paper, we prove Theorems 1, 2, 3, 4 using “only” the Log-Sobolev inequality. Since the hypercontractive inequality and the Log-Sobolev inequality are equivalent to each other and both have separate not-so-difficult proofs as well, whether one uses one or the other is, admittedly, splitting hairs. Still, another interesting aspect of this paper is that our proof approach is very different from all earlier proofs. We look at the first Fourier level of the function after a suitable random restriction and apply the Log-Sobolev inequality appropriately. The approach is, in our subjective opinion, more direct, natural, and less mysterious, though the overall proofs are not necessarily “easier”. The additional structural information implicit in these proofs might benefit further research. In Section 3, we describe the basic skeleton that is common to all our proofs and the main technical lemma, Lemma 20. The paper might have some benefit from expository perspective as all our proofs are uniformly built around the same skeleton.

Before proceeding to formal proofs, we illustrate here the underlying intuition and how it morally explains the KKL Theorem (translating the intuition into a formal proof takes some effort). We assume here that the reader is somewhat familiar with the area and the standard terminology. Let  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  be a balanced function and suppose all its influences are at most (say)  $\frac{1}{\sqrt{n}}$ . We hope to conclude that the total influence is then  $\Omega(\log n)$ . Suppose  $f$  has degree  $d$  (we are referring to the so-called average degree, but never mind). Consider a  $\frac{1}{d}$ -random restriction  $f_{\bar{J} \rightarrow z}$  of the function where each coordinate stays alive with probability  $\frac{1}{d}$  independently and denoting the set of alive coordinates as  $J$ , the coordinates in  $\bar{J} = [n] \setminus J$  are set to a uniformly random setting  $z$ . Since  $f$  has degree  $d$ , we expect that the restricted function  $f_{\bar{J} \rightarrow z}$  has constant Fourier weight at the first level and ideally, is even a dictatorship function (indeed, if the Fourier weight at the first level exceeds a certain threshold, a Boolean function is necessarily a dictatorship). Suppose, for the sake of illustration, that the restricted function  $f_{\bar{J} \rightarrow z}$  is always a dictatorship function. However, it could be the dictatorship of a different coordinate for different settings of  $z$ . Let  $A_j \subseteq \{0, 1\}^{\bar{J}}$  consist of those settings of  $z$  for which  $f_{\bar{J} \rightarrow z}$  is the dictatorship of coordinate  $j \in J$ . We note that the fractional size of  $A_j$ , denoted  $\mu(A_j)$ , is at most the influence of the coordinate  $j$  (why?) and hence  $\mu(A_j) \leq \frac{1}{\sqrt{n}}$  for all  $j \in J$ . Now we simply note that since the sets  $A_1, \dots, A_{|J|}$  are all polynomially small in size and form a partition of  $\{0, 1\}^{\bar{J}}$ , at least  $\frac{\log n}{n}$  fraction of the edges in the hypercube  $\{0, 1\}^{\bar{J}}$  are across some  $A_j$  and  $A_{j'}$  with  $j \neq j'$ . These edges, along with the fact that  $A_j$  and  $A_{j'}$  are restrictions leading to dictatorships of  $j$  and  $j'$  respectively, contribute  $\Omega(\log n)$  to the total influence of the function  $f$  as desired (why?)! We use here the standard isoperimetric result on the hypercube that for a small set  $A \subseteq \{0, 1\}^n$ , at least  $\frac{\log(1/\mu(A))}{n}$  fraction of hypercube edges incident on it, go outside of  $A$  (this is also a special case of the Log-Sobolev inequality, see Lemma 11).

## 2 Preliminaries

We denote  $[n] = \{1, 2, \dots, n\}$ . We write  $X \gtrsim Y$  to say that there exists an absolute constant  $c > 0$  such that  $X \geq c \cdot Y$ .

### 2.1 Standard Fourier Analysis

We consider the space of real-valued functions  $f : \{0, 1\}^n \rightarrow \mathbb{R}$ , equipped with the inner product  $\langle f, g \rangle = \mathbb{E}_{x \in_R \{0, 1\}^n} [f(x)g(x)]$ . Here and throughout the paper, we consider the uniform distribution over  $\{0, 1\}^n$ . It is well-known that the collection of functions  $\chi_S : \{0, 1\}^n \rightarrow \{-1, 1\}$ , one for each subset  $S \subseteq [n]$ , defined as  $\chi_S(x) = (-1)^{\oplus_{i \in S} x_i}$ , is an orthonormal basis w.r.t. the said inner product. Thus each function  $f : \{0, 1\}^n \rightarrow \mathbb{R}$  can be written uniquely as

$$f(x) = \sum_{S \subseteq [n]} \hat{f}(S) \chi_S(x), \quad \text{where } \hat{f}(S) = \langle f, \chi_S \rangle.$$

Since the basis  $\{\chi_S\}_{S \subseteq [n]}$  is orthonormal, one has the Plancherel/Parseval equality:

► **Fact 5.** For any  $f, g: \{0, 1\}^n \rightarrow \mathbb{R}$ , we have  $\langle f, g \rangle = \sum_{S \subseteq [n]} \hat{f}(S) \hat{g}(S)$ . Also

$$\langle f, f \rangle = \mathbb{E}_x [f(x)^2] = \|f\|_2^2 = \sum_{S \subseteq [n]} \hat{f}(S)^2.$$

We will also consider other  $L_p$  norms of functions for  $p \geq 1$  (mostly  $L_1$ -norm), similarly defined as  $\|f\|_p = (\mathbb{E}_x [|f(x)|^p])^{1/p}$ . It will be useful to consider the “Fourier weight” of a function on a given “level”.

► **Definition 6.** For integer  $d \geq 1$ , the level  $d$  Fourier weight of a function  $f: \{0, 1\}^n \rightarrow \mathbb{R}$  is  $W_{=d}[f] = \sum_{|S|=d} \hat{f}(S)^2$ . Also, its Fourier weight on the “chunk”  $d$  is  $W_{\approx d}[f] = \sum_{d \leq |S| < 2d} W_{=|S|}[f]$ .

For a noise parameter  $\varepsilon \in (0, 1)$ , the noise operator  $T_{1-\varepsilon}$  is defined as follows. For a function  $f: \{0, 1\}^n \rightarrow \mathbb{R}$ , the function  $T_{1-\varepsilon}f$  is

$$T_{1-\varepsilon}f(x) = \mathbb{E}_{y \sim_\varepsilon x} [f(y)],$$

where the input  $y$  is obtained from input  $x$  by resampling each coordinate of  $x$  with probability  $\varepsilon$  independently. It is well-known that the Fourier representation of  $T_{1-\varepsilon}f$  is

$$T_{1-\varepsilon}f = \sum_{S \subseteq [n]} (1-\varepsilon)^{|S|} \hat{f}(S) \chi_S.$$

## 2.2 Discrete Derivatives and Influences

For a coordinate  $i \in [n]$ , the discrete derivatives of  $f$  along the  $i^{\text{th}}$  direction is a function  $\partial_i f: \{0, 1\}^{n-1} \rightarrow \mathbb{R}$  defined as

$$\partial_i f(y) = f(x_{-i} = y, x_i = 1) - f(x_{-i} = y, x_i = 0).$$

► **Definition 7.** The  $L_p$ -influence of a coordinate  $i \in [n]$  is defined as  $I_i^p[f] = \|\partial_i f\|_p^p$ . The  $L_p$  total-influence is  $I^p[f] = \sum_{i=1}^n I_i^p[f]$ . We stress here that in the notation  $I_i^p[f]$  and  $I^p[f]$  herein, the “ $p$ ” is a super-script and not an exponent.

We will be concerned with only  $L_2$  and  $L_1$  influences. In the literature, the notion usually refers to  $L_2$ -influences, so in this case the superscript  $p$  is omitted, writing  $I_i[f] = I_i^2[f]$  and  $I[f] = I^2[f]$  for the individual and total influence respectively. We note that for Boolean functions, all the  $L_p$ -influences are equal. We will be concerned with the more general case of bounded functions, i.e. functions taking values in the interval  $[-1, 1]$ , and state our variants of Theorems 1, 2, 3, and 4 using  $L_1$ -influences instead. We remark that for bounded functions, one has  $I_i^p[f] \leq I_i^q[f]$  for  $p \geq q \geq 1$ . In particular and via Cauchy-Schwartz,  $I_i[f] \leq I_i^1[f] \leq \sqrt{I_i[f]}$ . Using the Fourier expansion of the discrete derivatives and Parseval equality gives the following standard formula for the total  $L_2$ -influence.

► **Fact 8.** For any  $f: \{0, 1\}^n \rightarrow \mathbb{R}$ , we have  $I[f] = 4 \sum_{S \subseteq [n]} |S| \hat{f}(S)^2$ . In particular, by an averaging argument, for any  $\varepsilon > 0$ ,  $\sum_{|S| \geq I[f]/\varepsilon} \hat{f}(S)^2 \leq \varepsilon$ .

### 2.3 Hypercontractive Inequality and Log-Sobolev Inequality

The hypercontractive inequality states that for each  $\varepsilon > 0$ , there is  $p > 2$  such that  $T_{1-\varepsilon}$  is a contraction from  $L_2$  to  $L_p$ , i.e. that  $\|T_{1-\varepsilon}f\|_p \leq \|f\|_2$  for any  $f: \{0,1\}^n \rightarrow \mathbb{R}$ . The inequality has an equivalent form (which is often times used) that does not involve the noise operator  $T_{1-\varepsilon}$ , and is instead concerned with bounded degree functions.

The degree of a function  $f$ , denoted  $\deg(f)$ , is the maximum of  $|S|$  over all  $S$  such that  $\hat{f}(S) \neq 0$ . The Bonami-Beckner hypercontractive inequality [2, 1] asserts that the  $L_p$ -norm and the  $L_2$ -norm of a low-degree function are comparable. More precisely, for any  $f: \{0,1\}^n \rightarrow \mathbb{R}$  and any  $p > 2$ ,

► **Theorem 9.**  $\|f\|_p \leq (p-1)^{\deg(f)/2} \|f\|_2$ .

To motivate the Log-Sobolev inequality and its relationship to the hypercontractive inequality, let us rewrite the above as

$$\deg(f) \geq \frac{2}{\log(p-1)} \log \left( \frac{\|f\|_p}{\|f\|_2} \right). \quad (1)$$

Instead of looking at the maximal degree of a non-zero monomial that appears in  $f$ , one may consider the average degree of  $f$ , defined as  $\sum_S |S| \hat{f}(S)^2$ , where the weight given to a characters  $S$  equals the squared Fourier coefficient  $\hat{f}(S)^2$ . When  $f$  is  $\{-1,1\}$ -valued, the squared Fourier coefficients sum up to 1, giving a probability distribution over them, explaining the term “average degree”. As noted, the average degree is same as the total influence  $I[f]$  (up to the factor 4). The Log-Sobolev inequality, established by Gross [12], can be seen as the limiting case of the above inequality as  $p \rightarrow 2$  and replacing the degree by average degree (see [12], [17, Chapter 10.1] and [17, Pages 319-320] for the equivalence between the two inequalities and also separate inductive proofs). Towards stating this inequality, one needs the notion of entropy of a non-negative function  $h: \{0,1\}^n \rightarrow [0, \infty)$ :

$$\text{Ent}(h) := \mathbb{E}_x [h(x)] \log \left( \frac{1}{\mathbb{E}_x [h(x)]} \right) - \mathbb{E}_x \left[ h(x) \log \left( \frac{1}{h(x)} \right) \right],$$

with the convention that  $0 \log(1/0) = 0$ . The Log-Sobolev inequality is (note that the entropy is of the non-negative function  $f^2$ ):

► **Theorem 10.** For any  $f: \{0,1\}^n \rightarrow \mathbb{R}$ , we have  $I[f] \geq \frac{1}{2} \text{Ent}(f^2)$ .

A simple corollary of this inequality, when  $f: \{0,1\}^n \rightarrow \{0,1\}$  is Boolean, is below. This is also known as the standard isoperimetric inequality for the Boolean hypercube.

► **Lemma 11.** For any  $f: \{0,1\}^n \rightarrow \{0,1\}$ ,  $\beta = \mathbb{E}[f] \leq \frac{1}{2}$ , we have  $I[f] \geq \frac{1}{2} \beta \log(1/\beta)$ .

It will be more convenient for us to use the following easy consequence of the Log-Sobolev inequality.

► **Lemma 12.** There exists an absolute constant  $K > 0$ , such that for any  $f: \{0,1\}^n \rightarrow [-1,1]$ , we have

$$I[f] \gtrsim \|f\|_2^2 \log \left( \frac{1}{\|f\|_2^2} \right) - K \cdot \|f\|_1^{\frac{1}{2}} \|f\|_2.$$

**Proof.** By Theorem 10,  $I[f] \gtrsim \text{Ent}(f^2)$ , so it is enough to show that the entropy of  $f^2$  is at least the right hand side. Indeed, the first term in the definition of the entropy is precisely  $\|f\|_2^2 \log(1/\|f\|_2^2)$ . The second term is (using Cauchy-Schwarz and that  $t^2 \log^2(1/t^2) \lesssim |t|$  for  $t \in [-1,1]$ )

$$\begin{aligned} \mathbb{E}_x \left[ f(x)^2 \log \left( \frac{1}{f(x)^2} \right) \right] &\leq \sqrt{\mathbb{E}_x \left[ f(x)^2 \log^2 \left( \frac{1}{f(x)^2} \right) \right] \mathbb{E}_x [f(x)^2]} \lesssim \sqrt{\mathbb{E}_x [|f(x)|] \mathbb{E}_x [f(x)^2]} \\ &= \|f\|_1^{\frac{1}{2}} \|f\|_2. \end{aligned}$$

◀

## 2.4 Random Restrictions

Let  $J \subseteq [n]$  be a subset of coordinates thought of as “alive” and coordinates in  $\bar{J} = [n] \setminus J$  thought of as “restricted”. Given a function  $f: \{0, 1\}^n \rightarrow \mathbb{R}$  and a setting  $z \in \{0, 1\}^{\bar{J}}$ , we denote by  $f_{\bar{J} \rightarrow z}$ , the restriction of  $f$  to the domain  $z \times \{0, 1\}^J$ . More precisely,  $f_{\bar{J} \rightarrow z}: \{0, 1\}^J \rightarrow \mathbb{R}$  is defined as  $f_{\bar{J} \rightarrow z}(y) = f(x_{\bar{J}} = z, x_J = y)$ . The following standard fact gives the Fourier coefficients of the restricted function:

► **Fact 13.** For any  $T \subseteq J$ , we have  $\hat{f}_{\bar{J} \rightarrow z}(T) = \sum_{S \subseteq \bar{J}} \hat{f}(S \cup T) \chi_S(z)$ .

For a parameter  $\delta > 0$ , a  $\delta$ -random restriction is the function  $f_{\bar{J} \rightarrow z}$  after choosing  $J$  to be a random subset of  $[n]$  in which each  $j \in [n]$  is included with probability  $\delta$  independently and choosing  $z \in \{0, 1\}^{\bar{J}}$  uniformly. Using Fact 13 and Parseval, one can easily compute the expected squared Fourier coefficient of a random restriction and then the expected level  $d$  Fourier weight.

► **Fact 14.** Let  $f: \{0, 1\}^n \rightarrow \mathbb{R}$  and  $T \subseteq J$ . Then  $\mathbb{E}_z [|\hat{f}_{\bar{J} \rightarrow z}(T)|^2] = \sum_{S \subseteq \bar{J}} \hat{f}(S \cup T)^2$ .

► **Fact 15.** Let  $f: \{0, 1\}^n \rightarrow \mathbb{R}$ ,  $d \geq 1$  be an integer, and  $\delta \in [0, 1]$ . Let  $f_{\bar{J} \rightarrow z}$  denote the  $\delta$ -random restriction. Then

$$\mathbb{E}_{J, z} [W_{=d}[f_{\bar{J} \rightarrow z}]] = \sum_S \hat{f}(S)^2 \cdot \Pr_J[|J \cap S| = d].$$

## 3 A Basic Argument towards the KKL Theorem

In this section, we prove the lemma below. It proves the KKL Theorem in the special case when the function  $f: \{0, 1\}^n \rightarrow [0, 1]$  has a constant fraction of its Fourier weight on some “chunk”. Alternately, it proves the KKL Theorem at a loss of  $\log \log n$  factor. More importantly, the proof illustrates the basic approach underlying all the subsequent proofs.

► **Lemma 16.** Let  $f: \{0, 1\}^n \rightarrow [0, 1]$  be a function and  $d \geq 1$  be an integer. Then there exists a coordinate  $i \in [n]$  such that  $I_i^1[f] \gtrsim \frac{\log n}{n} W_{\approx d}[f]$ .

We make some remarks before proceeding to the proof. Firstly, we note that the lemma holds for bounded functions and with respect to the  $L_1$ -influences. Secondly, we note that if a constant fraction of the Fourier weight is on some chunk, i.e. if for some  $d$ ,  $W_{\approx d}[f] \gtrsim \text{var}(f)$ , then there is a coordinate  $i \in [n]$  with  $I_i^1[f] \gtrsim \frac{\log n}{n} \text{var}(f)$ , proving the KKL Theorem. Thirdly, we note that it proves the KKL Theorem at a loss of factor  $\log \log n$  as follows. We may assume that  $I[f] \leq \frac{1}{2} \log n \text{var}(f)$ . Since  $\text{var}(f) = \sum_{1 \leq |S|} \hat{f}(S)^2$  and  $I[f] = 4 \sum_S |S| \hat{f}(S)^2$ , by Markov’s inequality, we have

$$\sum_{1 \leq |S| \leq \log n} \hat{f}(S)^2 \geq \frac{1}{2} \text{var}(f).$$

Thus, by partitioning the interval  $[1, \log n]$  into  $\log \log n$  dyadic intervals  $\bigcup_{k=0}^{\log \log n} [2^k, 2^{k+1})$ , it follows that there is some  $1 \leq d \leq \log n$  such that  $W_{\approx d} \gtrsim \frac{\text{var}(f)}{\log \log n}$ . Hence by the lemma, there is a coordinate  $i \in [n]$  such that  $I_i^1[f] \gtrsim \frac{\log n}{n} W_{\approx d}[f] \gtrsim \frac{\log n}{n} \frac{\text{var}(f)}{\log \log n}$ .

### 3.1 Proof of Lemma 16

We now prove Lemma 16. The proof formalizes the intuition described at the end of the introductory section. One begins by considering a  $\frac{1}{d}$ -random restriction of the function, notes that the expected Fourier weight at the first level of the restricted function is at least  $W_{\approx d}[f]$ , and then one examines the coefficients at the first level and applies Log-Sobolev appropriately.

#### Weight on the First Level after Random Restriction

Let  $f_{\bar{J} \rightarrow z}$  be a  $\frac{1}{d}$ -random restriction,  $J$  being the set of coordinates left alive. We have by Fact 15 that

$$\mathbb{E}_{J,z} [W_{=1}[f_{\bar{J} \rightarrow z}]] \geq \sum_{d \leq |S| < 2d} \hat{f}(S)^2 \cdot \Pr_J[|S \cap J| = 1] \gtrsim W_{\approx d}[f], \quad (2)$$

where we used the simple fact that for any set  $S$  with  $d \leq |S| < 2d$ , the probability it intersects  $J$  in a single element is constant. For the rest of the argument, we fix some  $J \subseteq [n]$  such that (it exists due to Equation (2))

$$\mathbb{E}_z [W_{=1}[f_{\bar{J} \rightarrow z}]] \gtrsim W_{\approx d}[f]. \quad (3)$$

#### Relating First Level Coefficients after Restriction and Influences of $f$

We now consider the first level coefficients of the restricted function  $f_{\bar{J} \rightarrow z}$  and somehow relate them to the influences of the original function  $f$ . We note that  $J$  is the set of alive coordinates. For each  $j \in J$ , define a function  $g_j: \{0, 1\}^{\bar{J}} \rightarrow \mathbb{R}$  by  $g_j(z) = \hat{f}_{\bar{J} \rightarrow z}(\{j\})$ . That is,  $g_j(z)$  is the  $j^{\text{th}}$  coefficient of the first level (= linear part) of the restricted function. By definition,  $W_{=1}[f_{\bar{J} \rightarrow z}] = \sum_{j \in J} g_j(z)^2$ . Let  $p_j = \|g_j\|_2^2 = \mathbb{E}_z [g_j(z)^2]$ . For the sake of future reference, let  $q_j = \|g_j\|_1$ . Thus (3) can be re-stated as

$$\mathbb{E}_z [W_{=1}[f_{\bar{J} \rightarrow z}]] = \sum_{j \in J} p_j \gtrsim W_{\approx d}[f]. \quad (4)$$

Since  $f$  is bounded, so is its restriction, and hence  $|g_j(z)| \leq 1$  for every  $z, j$ .

► **Lemma 17.**  $p_j = \|g_j\|_2^2$  and  $q_j = \|g_j\|_1$  satisfy

- $q_j = \|g_j\|_1 \leq \frac{1}{2} \cdot I_j^1[f]$ .
- $p_j = \|g_j\|_2^2 \leq \frac{1}{4} \cdot I_j[f]$ .
- $p_j \leq q_j \leq \sqrt{p_j}$ .

**Proof.** The third item is because of the boundedness  $|g_j(z)| \leq 1$  and Cauchy-Schwartz. Towards the first two items, we note that

$$g_j(z) = \hat{f}_{\bar{J} \rightarrow z}(\{j\}) = \mathbb{E}_y [f(z, y) \chi_{\{j\}}(y_j)] = \mathbb{E}_{y-j} \left[ \frac{f(z, y_{-j}, y_j = 0) - f(z, y_{-j}, y_j = 1)}{2} \right].$$

Taking expectation over  $z$  gives (and using Cauchy-Schwartz in the second case)

$$\begin{aligned}\|g_j\|_1 &= \mathbb{E}_z [|g_j(z)|] \leq \mathbb{E}_{z, y-j} \left[ \left| \frac{f(z, y-j, y_j=0) - f(z, y-j, y_j=1)}{2} \right| \right] = \frac{1}{2} \cdot I_j^1[f]. \\ \|g_j\|_2^2 &= \mathbb{E}_z [|g_j(z)|^2] \leq \mathbb{E}_{z, y-j} \left[ \left| \frac{f(z, y-j, y_j=0) - f(z, y-j, y_j=1)}{2} \right|^2 \right] = \frac{1}{4} \cdot I_j[f]. \quad \blacktriangleleft\end{aligned}$$

Summing the previous inequality over all  $j \in J$ , we conclude that:

► **Lemma 18.**  $\sum_{j \in J} q_j = \sum_{j \in J} \|g_j\|_1 \leq \frac{1}{2} \cdot I^1[f]$ .

The following lower bound on  $I[f]$  is a key observation.

► **Lemma 19.**  $\sum_{j \in J} I[g_j] \leq I[f]$ .

**Proof.** We lower bound  $I[f]$  by  $\sum_{i \in \bar{J}} I_i[f]$ . Fix some  $i \in \bar{J}$  for now. As before,  $z$  and  $y$  denote the inputs on the parts  $\bar{J}$  and  $J$  respectively.

$$I_i[f] = \mathbb{E}_{z, y} [|f(z, y) - f(z \oplus e_i, y)|^2] = \mathbb{E}_z [\|f_{\bar{J} \rightarrow z} - f_{\bar{J} \rightarrow z \oplus e_i}\|_2^2].$$

By Parseval, we express the squared norm in terms of Fourier coefficients and then lower bound by considering only coefficients of size one.

$$I_i[f] = \mathbb{E}_z \left[ \sum_{T \subseteq J} |\hat{f}_{\bar{J} \rightarrow z}(T) - \hat{f}_{\bar{J} \rightarrow z \oplus e_i}(T)|^2 \right] \geq \mathbb{E}_z \left[ \sum_{j \in J} |\hat{f}_{\bar{J} \rightarrow z}(\{j\}) - \hat{f}_{\bar{J} \rightarrow z \oplus e_i}(\{j\})|^2 \right].$$

The latter are simply  $g_j(z)$  and  $g_j(z \oplus e_i)$  by definition and hence

$$I_i[f] \geq \sum_{j \in J} \mathbb{E}_z [|g_j(z) - g_j(z \oplus e_i)|^2] = \sum_{j \in J} I_i[g_j].$$

Summing over  $i \in \bar{J}$  gives

$$I[f] \geq \sum_{i \in \bar{J}} I_i[f] \geq \sum_{i \in \bar{J}} \sum_{j \in J} I_i[g_j] = \sum_{j \in J} \sum_{i \in \bar{J}} I_i[g_j] = \sum_{j \in J} I[g_j]. \quad \blacktriangleleft$$

## The Main Argument

Our main argument tries to obtain a lower bound on  $I[f]$  as follows. Using Lemma 19 and the Log-Sobolev Lemma 12,

$$I[f] \geq \sum_{j \in J} I[g_j] \geq \sum_{j \in J} (p_j \log(1/p_j) - K\sqrt{q_j} \cdot \sqrt{p_j}).$$

Using Cauchy-Schwartz, we get

$$I[f] \geq \sum_{j \in J} p_j \log(1/p_j) - K \sqrt{\sum_{j \in J} q_j} \cdot \sqrt{\sum_{j \in J} p_j}.$$

By Lemma 18,  $\sum_{j \in J} q_j \leq I^1[f]$ , so we get our main technical inequality

$$I[f] \geq \sum_{j \in J} p_j \log(1/p_j) - K \sqrt{I^1[f]} \cdot \sqrt{\sum_{j \in J} p_j}. \quad (5)$$



We recall that  $p_j \leq \frac{1}{2} I_j^1[f]$  by Lemma 17. Letting  $W := \sum_{j \in J} p_j = \mathbb{E}_z[W_{=1}[f_{\bar{J} \rightarrow z}]]$ , we rewrite this inequality, for future reference, as below. We note that in application,  $J$  is the subset of alive coordinates after a random restriction. In our proof of KKL and Friedgut Theorems, the set  $J$  is fixed so as to maximize the expected first level Fourier weight. In the proof of Talagrand Theorem, we average over the choice of  $J$  as well.

► **Lemma 20.** *Let  $f: \{0, 1\}^n \rightarrow [0, 1]$  be a function and  $J \subseteq [n]$ . Then*

$$I[f] \geq \log \left( \frac{1}{\max_{j \in J} I_j^1[f]} \right) \cdot W - K \sqrt{I^1[f]} \cdot \sqrt{W},$$

$$W = \mathbb{E}_z[W_{=1}[f_{\bar{J} \rightarrow z}]] = \sum_{S \subseteq [n], |S \cap J|=1} \hat{f}(S)^2.$$

The proof of Lemma 16 is now completed immediately. We may assume that for all coordinates  $i \in [n]$ ,  $I_i^1[f] \leq \frac{\log n}{n}$ ,  $W_{\approx d}[f] \leq \frac{1}{\sqrt{n}}$  as otherwise we are done already. This implies that the total  $L_1$ -influence  $I^1[f] \leq \log n W_{\approx d}[f]$ . Lemma 20 (= Equation (5)) then gives (the log-factor therein is at least  $\frac{1}{2} \log n$  since all  $L_1$ -influences are at most  $\frac{1}{\sqrt{n}}$ )

$$I[f] \geq \frac{1}{2} \log n \cdot W - K \sqrt{\log n \cdot W_{\approx d}[f]} \cdot \sqrt{W}, \quad W \gtrsim W_{\approx d}[f].$$

Clearly, the first term above dominates the second, giving  $I[f] \geq \frac{1}{4} \log n \cdot W \gtrsim \log n \cdot W_{\approx d}[f]$ , implying now that there is a coordinate with in fact  $L_2$ -influence  $\gtrsim \frac{\log n}{n} W_{\approx d}[f]$ .

## 4 The KKL Theorem

We now prove the KKL Theorem, stated below for a bounded function, with respect to  $L_1$ -influences, and in a slightly different form.

► **Theorem 21.** *There exists an absolute constant  $c > 0$  such that the following holds. Let  $f: \{0, 1\}^n \rightarrow [0, 1]$  be a function. Then either  $I^1[f] \geq c \cdot \log n \cdot \text{var}(f)$ , or there is a coordinate  $i \in [n]$  such that  $I_i^1[f] \geq \frac{1}{\sqrt{n}}$ .*

It will be more convenient for us to prove a dimension-independent version of the KKL Theorem below. It is easily seen to imply the statement above.

► **Theorem 22.** *There exists an absolute constant  $C > 0$  such that the following holds. Let  $f: \{0, 1\}^n \rightarrow [0, 1]$  be a function. Then there is a coordinate  $i \in [n]$  such that  $I_i^1[f] \geq 2^{-C \cdot \frac{I^1[f]}{\text{var}(f)}}$ .*

In the proof of Lemma 16, we only “utilized” Fourier weight from a single chunk of Fourier coefficients, i.e. those of size in the range  $[d, 2d)$ , and this led to a loss of factor  $\log \log n$  if used towards the KKL Theorem. In this section, we show how to utilize and combine the Fourier weight from multiple chunks, avoiding this loss. The idea is to “partition”  $f$  into chunks as  $f = \hat{f}(\emptyset) + \sum_{d=2^k, k \geq 0} h_d^*$ , apply the main technical inequality (5) to each chunk  $h_d^*$ , and then “sum up”. A natural way to partition is to let  $h_d^* = \sum_{d \leq |S| < 2d} \hat{f}(S) \chi_S$ . The problem with this approach however is that the chunk functions  $h_d^*$  as here are not necessarily bounded functions and the earlier arguments cannot be applied directly. To get around this, we instead consider a soft notion of chunks,  $f \approx \hat{f}(\emptyset) + \sum_{d=2^k, k \geq 0} h_d$ , that behaves similarly, that is

$$\sum_d \text{var}(h_d) = \Theta(\text{var}(f)), \quad \sum_d I_i[h_d] = \Theta(I_i[f]), \quad \sum_d I[h_d] = \Theta(I[f]),$$

and in addition, preserves boundedness and the  $L_1$ -influences of each soft chunk  $h_d$  are bounded by those of the original function!

### 4.1 Soft Chunks

► **Definition 23.** Let  $f: \{0, 1\}^n \rightarrow [0, 1]$  be a function and let  $d \geq 1$  be integer (thought of as a power of 2). The soft chunk of  $f$  of degree  $d$  is given by the function  $h_d: \{0, 1\}^n \rightarrow [0, 1]$  defined by  $h_d = (T_{1-\frac{1}{2d}} - T_{1-\frac{1}{d}})f$ .

The following lemma summarizes the useful properties of soft chunks (the proof appears in Appendix A.1). We point out, in particular, that the  $L_1$ -influences of the soft chunk are upper bounded by those of the original function (up to a factor 2).

► **Lemma 24.** Let  $f: \{0, 1\}^n \rightarrow [0, 1]$  and for integer  $d = 2^k, k \geq 0$ , let  $h_d = (T_{1-\frac{1}{2d}} - T_{1-\frac{1}{d}})f$  denote the soft chunk of  $f$  of degree  $d$ . Then (the sums are over  $d = 2^k, k \geq 0$  and  $i \in [n]$  is arbitrary)

- $h_d$  is bounded in  $[-1, 1]$ ,  $\hat{h}(\emptyset) = 0$ .
- $I_i^1[h_d] \leq 2I_i^1[f]$ .
- For any  $S \subseteq [n]$ ,  $d \leq |S| < 2d$ , we have  $|\hat{f}(S)| \lesssim |\hat{h}_d(S)| \leq |\hat{f}(S)|$ . In particular, we have lower bounds

$$\|h_d\|_2^2 \geq W_{\approx d}[h_d] \gtrsim W_{\approx d}[f], \quad \sum_d I_i[h_d] \gtrsim I_i[f], \quad \sum_d I[h_d] \gtrsim I[f].$$

- And the upper bounds,

$$\sum_d \|h_d\|_2^2 \leq \text{var}(f), \quad \sum_d I[h_d] \leq I[f].$$

For technical reasons, we will be able to “utilize” only those chunks that have a significant amount of Fourier weight, referred to as the good chunks. It will turn out that the good chunks still capture a constant fraction of the variance of  $f$ , so this will not be a problem. Towards this end, we have (proof appears in Appendix A.2)

► **Lemma 25.** Let

$$D_{\text{good}} := \left\{ d = 2^k, k \geq 0 \mid W_{\approx d}[f] \geq \frac{\text{var}(f)^2}{16 \cdot I^1[f]} \right\}.$$

Then

$$\sum_{d \in D_{\text{good}}} W_{\approx d}[f] \gtrsim \text{var}(f).$$

### 4.2 Proof of Theorem 22

Assume, for the sake of contradiction, that for all coordinates  $i \in [n]$ ,  $I_i^1[f] \leq 2^{-C \cdot \frac{I^1[f]}{\text{var}(f)}}$  where  $C$  is a large enough constant chosen later. Let  $h_d, d \in D_{\text{good}}$  be any good soft chunk. We recall that

- $h_d$  is a bounded function.
- Its  $L_1$ -influences are upper bounded by those of  $f$  up to a factor 2 (and hence also the total  $L_1$ -influence).
- $W_{\approx d}[h_d] \gtrsim W_{\approx d}[f] \geq \frac{\text{var}(f)^2}{16 \cdot I^1[f]}$ .

We apply Lemma 20 to the function  $h_d$ , considering  $\frac{1}{d}$ -random restriction, and letting  $J$  to be the subset of alive coordinates (fixed so as to maximize expected weight at first Fourier level). This yields the inequality

$$I[h_d] \geq \log \left( \frac{1}{\max_{j \in J} I_j^1[h_d]} \right) \cdot W - K \sqrt{I^1[h_d]} \cdot \sqrt{W}, \quad W \gtrsim W_{\approx d}[h_d] \gtrsim W_{\approx d}[f] \geq \frac{\text{var}(f)^2}{16 \cdot I^1[f]}.$$

Since the  $L_1$ -influences of  $h_d$  are bounded by those of  $f$ , in particular all of them at most  $2^{-C \cdot \frac{I^1[f]}{\text{var}(f)}}$ , we get

$$I[h_d] \geq C \cdot \frac{I^1[f]}{\text{var}(f)} \cdot W - K \sqrt{I^1[f]} \cdot \sqrt{W}, \quad W \gtrsim W_{\approx d}[f] \geq \frac{\text{var}(f)^2}{16 \cdot I^1[f]}.$$

It is easily seen that for a large enough constant  $C$ , the first term dominates the second term (this is why we considered only the good chunks) and thus

$$I[h_d] \gtrsim C \cdot \frac{I^1[f]}{\text{var}(f)} \cdot W_{\approx d}[f].$$

Now summing over all good  $d$  gives a contradiction:

$$I^1[f] \geq I[f] \geq \sum_{d \in D_{\text{good}}} I[h_d] \gtrsim C \cdot \frac{I^1[f]}{\text{var}(f)} \sum_{d \in D_{\text{good}}} W_{\approx d}[f] \gtrsim C \cdot \frac{I^1[f]}{\text{var}(f)} \cdot \text{var}(f) = C \cdot I^1[f].$$

We used Lemma 24 in the second step and Lemma 25 in the second-last step. Taking the constant  $C$  large enough gives a contradiction.

## 5 The Friedgut's Junta Theorem

Friedgut's Junta Theorem (restated below) is proved by a careful adjustment to the argument in the previous section.

► **Theorem 26.** *There is an absolute constant  $C > 0$  such that the following holds. For every function  $f: \{0, 1\}^n \rightarrow [0, 1]$  and for every  $\varepsilon > 0$ , there exists a function  $g: \{0, 1\}^n \rightarrow [0, 1]$  depending on at most  $2^{C \cdot I^1[f]/\varepsilon}$  variables such that  $\|f - g\|_2^2 \lesssim \varepsilon$ .*

We provide a proof sketch. While in the proof of the KKL Theorem, we may assume that all influences are small, this is not the case with Friedgut's Theorem. Here we “separate out” the set  $L$  of coordinates with “non-negligible” influence and apply the previous argument to the remaining set  $\bar{L} = [n] \setminus L$ . Towards this end, let

$$L = \left\{ i \mid I_i^1[f] \geq \tau := 2^{-C \cdot I^1[f]/\varepsilon} \right\}.$$

Clearly,  $|L| \leq \frac{I^1[f]}{\tau} \leq 2^{2C \cdot I^1[f]/\varepsilon}$ . Let  $g = \sum_{S \subseteq \bar{L}} \hat{f}(S) \chi_S$ . It is easily observed that

- $g$  depends only on the coordinates of  $\bar{L}$ .
- $g$  is also bounded in  $[0, 1]$  since  $g$  is simply the average of  $f$  over coordinates in  $\bar{L}$  and
- for the same reason,  $L_1$ -influences of  $g$  are bounded by those of  $f$ .

Let  $\varphi = f - g$ . We will show that  $\|\varphi\|_2^2 \lesssim \varepsilon$ . Clearly,  $\varphi$  is bounded in  $[-1, 1]$  and its  $L_1$  influences are also bounded by those of  $f$  up to a factor 2. We intend to apply the same argument used to prove the KKL Theorem to  $\varphi$ , except that all “action” happens only on the set of coordinates  $\bar{L}$ . More specifically:

- The “size” of any Fourier term is counted as  $|S \cap \bar{L}|$  instead of as  $|S|$ .
- For an integer  $d \geq 1$  (thought of as power of 2), the Fourier weight on the corresponding chunk is defined as

$$W_{\approx d}^{\bar{L}}[\varphi] := \sum_{d \leq |S \cap \bar{L}| < 2d} \hat{f}(S)^2.$$

## 26:12 Analytical Theorems via Random Restrictions and Log-Sobolev Inequality

- Towards defining the soft chunk  $h_d$  of  $\varphi$ , the noise operator is applied only to coordinates in  $\bar{L}$ . We denote this as

$$h_d = \left( T_{1-\frac{1}{2d}}^{\bar{L}} - T_{1-\frac{1}{d}}^{\bar{L}} \right) \varphi.$$

- In a random restriction, only coordinates in  $\bar{L}$  may stay alive. That is, a  $\frac{1}{d}$ -random restriction amounts to letting  $J$  to be a random subset of  $\bar{L}$  where every coordinate in  $\bar{L}$  is included with probability  $\frac{1}{d}$  and then the coordinates outside  $J$  (including those in  $L$ ) are set uniformly at random.
- Since  $J \subseteq \bar{L}$ , we have  $I_j^1[\varphi] \leq 2^{-C \cdot I^1[f]/\varepsilon}$  for all  $j \in J$ .

Modulo these considerations, we repeat the proof in Section 4.2. We apply Lemma 20 to the function  $h_d$ , considering  $\frac{1}{d}$ -random restriction, and letting  $J$  be the subset of alive coordinates (fixed so as to maximize expected weight at first Fourier level). This yields the inequality

$$I[h_d] \geq \log \left( \frac{1}{\max_{j \in J} I_j^1[h_d]} \right) \cdot W - K \sqrt{I^1[h_d]} \cdot \sqrt{W}, \quad W \gtrsim W_{\approx d}^{\bar{L}}[h_d] \gtrsim W_{\approx d}^{\bar{L}}[\varphi].$$

Since the  $L_1$ -influences of  $h_d$  are bounded by those of  $\varphi$  which are in turn bounded by those of  $f$  and those for coordinates in  $J \subseteq \bar{L}$  are at most  $2^{-C \cdot \frac{I^1[f]}{\varepsilon}}$ , we get

$$I[h_d] \geq C \cdot \frac{I^1[f]}{\varepsilon} \cdot W - K \sqrt{I^1[f]} \cdot \sqrt{W}, \quad W \gtrsim W_{\approx d}^{\bar{L}}[\varphi].$$

Let  $D_{\text{good}}$  be the subset of  $d = 2^k$  such that  $W_{\approx d}^{\bar{L}}[\varphi] \geq \frac{\varepsilon^2}{16I^1[f]}$  so that for such good  $d$  and for large enough constant  $C$ , the first term above dominates the second and we get

$$I[h_d] \gtrsim C \cdot \frac{I^1[f]}{\varepsilon} W_{\approx d}^{\bar{L}}[\varphi].$$

Now summing over all good  $d \in D_{\text{good}}$  gives:

$$I^1[f] \geq I^1[\varphi] \geq I[\varphi] \geq \sum_{d \in D_{\text{good}}} I[h_d] \gtrsim C \cdot \frac{I^1[f]}{\varepsilon} \cdot \sum_{d \in D_{\text{good}}} W_{\approx d}^{\bar{L}}[\varphi].$$

By Lemma 25 (applied to  $\varphi$ ), the last sum is at least  $\gtrsim \text{var}(\varphi)$  and we get  $\text{var}(\varphi) \lesssim \varepsilon$  as desired. An astute reader might object that the definition of the good soft chunks here seems different than that in Lemma 25, i.e. the threshold is set at  $\frac{\varepsilon^2}{16I^1[f]}$  instead of  $\frac{\text{var}(\varphi)^2}{16I^1[\varphi]}$  therein. However since  $I^1[\varphi] \leq I^1[f]$  and we could assume a priori that  $\text{var}(\varphi) \geq \varepsilon$  (otherwise we would already be done), this slight difference only works in our favor.

## 6 The Talagrand's Theorem

In this section, we prove Talagrand's Theorem, restated as Theorem 28 later. For now we prove the following weaker theorem to illustrate the main idea.

► **Theorem 27.** *Let any  $f: \{0, 1\}^n \rightarrow [0, 1]$  be a function and  $d \geq 1$  an integer (thought of as power of 2). Then one of these two conclusions holds:*

- (Case 1):  $\sum_{j \in [n]} \frac{I_j[f]}{\log(1/I_j^1[f])} \gtrsim \frac{d(W_{\approx d}[f])^2}{I[f]}.$
- (Case 2):  $\sum_{j \in [n]} \frac{I_j^1[f]}{\log(1/I_j^1[f])} \gtrsim dW_{\approx d}[f].$

We make a few remarks. On the left hand side of the inequalities, what appear in the numerators are the  $L_2$ -influences in Case 1 and  $L_1$ -influences in Case 2. This distinction will be important later. In both cases, in the denominator, it does not matter whether we write  $L_1$  or  $L_2$  influences since their logarithms are the same up to a factor 2 (since  $I_j[f] \leq I_j^1[f] \leq \sqrt{I_j[f]}$ ). If one pretends that all non-zero Fourier coefficients of  $f$  have size between  $d$  and  $2d$ , we have  $W_{\approx d}[f] = \text{var}(f)$  and  $I[f] = \Theta(d \cdot \text{var}(f))$  and we get  $\text{var}(f)$  on the right hand side in Case 1 and (even better)  $d \cdot \text{var}(f)$  in Case 2, giving Talagrand's Theorem.

We now prove Theorem 27. Consider a  $\frac{1}{d}$ -random restriction as in Section 3 letting  $J$  to be the set of coordinates alive. As therein, let  $g_j(z) = \hat{f}_{\bar{J} \rightarrow z}(\{j\})$ ,  $p_j = \|g_j\|_2^2$ ,  $q_j = \|g_j\|_1$ . Unlike therein however, we will not fix the set  $J$  and instead take expectation over its choice. Exactly as in Equation (5), we get

$$I[f] \geq \sum_{j \in J} p_j \log(1/p_j) - K \sum_{j \in J} \sqrt{q_j} \sqrt{p_j}.$$

We now divide into two cases depending on whether or not, on the right hand side, the first term dominates the second. It will be more convenient to do this after considering expectation over choice of  $J$ .

**Case 1:**  $\mathbb{E}_J \left[ \sum_{j \in J} p_j \log(1/p_j) \right] \geq 2 \cdot \mathbb{E}_J \left[ K \sum_{j \in J} \sqrt{q_j} \sqrt{p_j} \right]$ .

In this case, we get

$$I[f] \gtrsim \mathbb{E}_J \left[ \sum_{j \in J} p_j \log(1/p_j) \right].$$

Cauchy-Schwartz gives,

$$\mathbb{E}_J \left[ \sum_{j \in J} \frac{p_j}{\log(1/p_j)} \right] \cdot \mathbb{E}_J \left[ \sum_{j \in J} p_j \log(1/p_j) \right] \geq \left( \mathbb{E}_J \left[ \sum_{j \in J} p_j \right] \right)^2.$$

The second term is bounded by  $I[f]$  (as above) and on the right hand side we have,  $\mathbb{E}_J \left[ \sum_{j \in J} p_j \right] \gtrsim W_{\approx d}[f]$ . This gives

$$\mathbb{E}_J \left[ \sum_{j \in J} \frac{p_j}{\log(1/p_j)} \right] \gtrsim \frac{(W_{\approx d}[f])^2}{I[f]}.$$

Replacing  $p_j$  by its upper bound  $I_j[f]$  in the numerator and its upper bound  $I_j^1[f]$  in the denominator, and noting that each coordinate appears in  $J$  with probability  $\frac{1}{d}$ , gives the desired inequality

$$\sum_{j \in [n]} \frac{I_j[f]}{\log(1/I_j^1[f])} \gtrsim \frac{d(W_{\approx d}[f])^2}{I[f]}.$$

**Case 2:**  $\mathbb{E}_J \left[ \sum_{j \in J} \sqrt{q_j} \sqrt{p_j} \right] \gtrsim \mathbb{E}_J \left[ \sum_{j \in J} p_j \log(1/p_j) \right].$

In this case, Cauchy-Schwartz gives

$$\begin{aligned} \mathbb{E}_J \left[ \sum_{j \in J} \frac{q_j}{\log(1/p_j)} \right] \cdot \mathbb{E}_J \left[ \sum_{j \in J} p_j \log(1/p_j) \right] &\geq \left( \mathbb{E}_J \left[ \sum_{j \in J} \sqrt{q_j} \sqrt{p_j} \right] \right)^2 \\ &\gtrsim \left( \mathbb{E}_J \left[ \sum_{j \in J} p_j \log(1/p_j) \right] \right)^2. \end{aligned}$$

Canceling  $\mathbb{E}_J \left[ \sum_{j \in J} p_j \log(1/p_j) \right]$  from both sides gives

$$\mathbb{E}_J \left[ \sum_{j \in J} \frac{q_j}{\log(1/p_j)} \right] \gtrsim \mathbb{E}_J \left[ \sum_{j \in J} p_j \log(1/p_j) \right] \geq \mathbb{E}_J \left[ \sum_{j \in J} p_j \right].$$

As before, the right hand side is  $\gtrsim W_{\approx d}[f]$ , and  $q_j, p_j$  are upper bounded by  $I_j^1[f]$ , and each coordinate appears in  $J$  with probability  $\frac{1}{d}$ . This gives the desired inequality

$$\sum_{j \in [n]} \frac{I_j^1[f]}{\log(1/I_j^1[f])} \gtrsim d W_{\approx d}[f].$$

## 6.1 Talagrand's Theorem by Combining Chunks: First Attempt

We (re-)state Talagrand's Theorem below.

► **Theorem 28.** *For any  $f: \{0, 1\}^n \rightarrow [0, 1]$ , we have  $\sum_{j \in [n]} \frac{I_j^1[f]}{\log(1/I_j^1[f])} \gtrsim \text{var}(f)$ .*

We attempt to prove this result by splitting

$$f = \hat{f}(\emptyset) + \sum_{d=2^k, k \geq 0} h_d,$$

where  $h_d = \sum_{d \leq |S| < 2d} \hat{f}(S) \chi_S$  are the chunks of  $f$ . The strategy is to apply Theorem 27 to each chunk  $h_d$  separately and “sum up” or “combine” the outcomes. A crucial observation is that the  $L_2$ -influences indeed sum up, that is

$$I_i[f] = \sum_d I_i[h_d].$$

This strategy (almost) works with a careful consideration of whether the Case 1 or the Case 2 applies for different chunks. The catch, as before, is that the chunks  $h_d$  are not necessarily bounded functions and their  $L_1$ -influences might not be under control. To get around this issue, we instead work with the soft chunks as before and the full proof is completed in the next sub-section. For now, we pretend that the chunks  $h_d$  are bounded functions and see how the proof proceeds. We also pretend that the  $L_1$ -influences of  $h_d$  are upper bounded by those of  $f$  (both these conditions do hold when soft chunks are considered!).

We apply Theorem 27 to  $h_d$ . Noting that  $W_{\approx d}[h_d] \geq W_{\approx d}[f]$ ,  $I[h_d] = \Theta(d \cdot W_{\approx d}[f])$ , and that  $L_1$ -influences of  $h_d$  are upper bounded by those of  $f$ , we conclude that for every  $d = 2^k, k \geq 0$ , one of these conclusions holds (perhaps both conclusions hold and if so, we pick one arbitrarily):

- (Case 1):  $\sum_{j \in [n]} \frac{I_j[h_d]}{\log(1/I_j^1[f])} \gtrsim W_{\approx d}[f]$ . Let  $D'$  be the set of such  $d$ .
- (Case 2):  $\sum_{j \in [n]} \frac{I_j^1[f]}{\log(1/I_j^1[f])} \gtrsim d W_{\approx d}[f]$ . Let  $D''$  be the set of such  $d$ .

Now we complete the proof as follows. Since  $\text{var}(f) = \sum_{d \in D'} W_{\approx d}[f] + \sum_{d \in D''} W_{\approx d}[f]$ , either of the two sums is at least  $\frac{1}{2} \text{var}(f)$ . If the first sum is, then (crucially using the fact that  $L_2$ -influences sum up)

$$\sum_{j \in [n]} \frac{I_j[f]}{\log(1/I_j^1[f])} \geq \sum_{j \in [n]} \sum_{d \in D'} \frac{I_j[h_d]}{\log(1/I_j^1[f])} = \sum_{d \in D'} \sum_{j \in [n]} \frac{I_j[h_d]}{\log(1/I_j^1[f])} \gtrsim \sum_{d \in D'} W_{\approx d} \gtrsim \text{var}(f),$$

as desired. Otherwise, we may assume  $\sum_{d \in D''} W_{\approx d}[f] \geq \frac{1}{2} \text{var}(f)$ . Since  $d$  ranges only over powers of 2, it follows that there is some  $d \in D''$  such that  $d W_{\approx d}[f] \gtrsim \text{var}(f)$  (why!). Using this particular choice of  $d$  in the Case 2 above, we get as desired

$$\sum_{j \in [n]} \frac{I_j^1[f]}{\log(1/I_j^1[f])} \gtrsim \text{var}(f).$$

## 6.2 Talagrand's Theorem by Combining Soft Chunks

We now complete the proof of Talagrand's Theorem 28. We carry out the same proof as in the previous sub-section, except that we use the soft chunks  $h_d = (T_{1-\frac{1}{2d}} - T_{1-\frac{1}{d}})f$ . It holds that  $W_{\approx d}[h_d] = \Theta(W_{\approx d}[f])$ . However one place we need to be careful about is that we required that  $I[h_d] = \Theta(d W_{\approx d}[f])$ . This need not be true in general. Hence we restrict ourselves to only those  $d \in D_{\text{good}}$  for which this condition holds. The lemma below shows that there is still a constant fraction of variance on these good chunks and this is enough to complete the proof (the sets  $D'$  and  $D''$  above are subsets of  $D_{\text{good}}$  now).

► **Lemma 29.** *Let*

$$D_{\text{good}} = \left\{ d = 2^k, k \geq 0 \mid d W_{\approx d}[f] \geq \frac{1}{40} I[h_d] \right\}.$$

*Then*

$$\sum_{d \in D_{\text{good}}} W_{\approx d}[f] \geq \frac{1}{2} \text{var}(f).$$

**Proof.** As we will see, it suffices to show that  $\sum_d \frac{I[h_d]}{d} \leq 20 \text{var}(f)$ . To see that, as  $\text{var}(f) = \sum_{S \neq \emptyset} \hat{f}(S)^2$ , it is enough to show that for each  $S \neq \emptyset$ , the term  $\hat{f}(S)^2$  appears in the sum  $\sum_d \frac{I[h_d]}{d}$  with a multiplicative factor of at most 20. Note that this factor is

$$|S| \sum_d \frac{1}{d} \left( \left(1 - \frac{1}{2d}\right)^{|S|} - \left(1 - \frac{1}{d}\right)^{|S|} \right)^2.$$

We analyze the contribution from  $d \leq |S|$  and  $d > |S|$  separately, showing that each one of them contributes at most  $\frac{10}{|S|}$ . Let  $k$  be such that  $2^k \leq |S| < 2^{k+1}$ . The first part is bounded as

$$\sum_{d \leq |S|} \frac{1}{d} \left(1 - \frac{1}{2d}\right)^{2|S|} \leq \sum_{d \leq |S|} \frac{1}{d} e^{-|S|/d} \leq \sum_{j=0}^k \frac{1}{2^j} e^{-2^{k-j}} \leq 2^{-k} \sum_{\ell=0}^{\infty} 2^{\ell} e^{-2^{\ell}} \leq 5 \cdot 2^{-k} \leq \frac{10}{|S|}.$$



The second part is bounded as (we approximate  $1 - r\alpha \leq (1 - \alpha)^r$  in this range).

$$\sum_{d > |S|} \frac{1}{d} \left( 1 - \left( 1 - \frac{1}{d} \right)^{|S|} \right)^2 \leq \sum_{d > |S|} \frac{1}{d} \leq \sum_{r=k+1} \frac{1}{2^r} = \frac{2}{2^{k+1}} \leq \frac{2}{|S|}.$$

This shows that  $\sum_d \frac{I[h_d]}{d} \leq 20 \operatorname{var}(f)$ . Now we complete the proof of the lemma as:

$$\begin{aligned} \sum_{d \in D_{\text{good}}} W_{\approx d}[f] &= \sum_d W_{\approx d}[f] - \sum_{d \notin D_{\text{good}}} W_{\approx d}[f] \\ &\geq \operatorname{var}(f) - \frac{1}{40} \sum_{d \notin D_{\text{good}}} \frac{I[h_d]}{d} \\ &\geq \operatorname{var}(f) - \frac{1}{40} \cdot 20 \cdot \operatorname{var}(f) \geq \frac{\operatorname{var}(f)}{2}. \end{aligned}$$

---

## References

- 1 William Beckner. Inequalities in Fourier analysis. *Annals of Mathematics*, 102:159–182, 1975.
- 2 Aline Bonami. étude des coefficients de Fourier des fonctions de  $l^p(g)$ . *Annales de l'Institut Fourier*, 20(2):335–402, 1970. doi:10.5802/aif.357.
- 3 Shuchi Chawla, Robert Krauthgamer, Ravi Kumar, Yuval Rabani, and D. Sivakumar. On the hardness of approximating multicut and sparsest-cut. *Computational Complexity*, 15(2):94–114, 2006. doi:10.1007/s00037-006-0210-9.
- 4 Nikhil R. Devanur, Subhash Khot, Rishi Saket, and Nisheeth K. Vishnoi. Integrality gaps for sparsest cut and minimum linear arrangement problems. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing, Seattle, WA, USA, May 21-23, 2006*, pages 537–546, 2006.
- 5 Irit Dinur and Ehud Friedgut. Intersecting families are essentially contained in juntas. *Combinatorics, Probability & Computing*, 18(1-2):107–122, 2009. doi:10.1017/S0963548308009309.
- 6 Irit Dinur and Samuel Safra. On the hardness of approximating minimum vertex cover. *Annals of mathematics*, pages 439–485, 2005.
- 7 Ronen Eldan and Renan Gross. Stability of talagrand’s influence inequality. *CoRR*, abs/1909.12067, 2019. arXiv:1909.12067.
- 8 Dvir Falik and Alex Samorodnitsky. Edge-isoperimetric inequalities and influences. *Combinatorics, Probability & Computing*, 16(5):693–712, 2007.
- 9 Ehud Friedgut. Boolean functions with low average sensitivity depend on few coordinates. *Combinatorica*, 18(1):27–35, 1998.
- 10 Ehud Friedgut and Gil Kalai. Every monotone graph property has a sharp threshold. *Proceedings of the American mathematical Society*, 124(10):2993–3002, 1996.
- 11 Dmitry Gavinsky, Julia Kempe, Iordanis Kerenidis, Ran Raz, and Ronald de Wolf. Exponential separation for one-way quantum communication complexity, with applications to cryptography. *SIAM J. Comput.*, 38(5):1695–1708, 2008. doi:10.1137/070706550.
- 12 Leonard Gross. Logarithmic sobolev inequalities. *American Journal of Mathematics*, 97(4):1061–1083, 1975. URL: <http://www.jstor.org/stable/2373688>.
- 13 J. Kahn, G. Kalai, and N. Linial. The influence of variables on Boolean functions. In IEEE, editor, *29th annual Symposium on Foundations of Computer Science, October 24–26, 1988, White Plains, New York*, pages 68–80, pub-IEEE:adr, 1988. IEEE Computer Society Press.
- 14 Subhash Khot and Oded Regev. Vertex cover might be hard to approximate to within 2-epsilon. *J. Comput. Syst. Sci.*, 74(3):335–349, 2008. doi:10.1016/j.jcss.2007.06.019.
- 15 Robert Krauthgamer and Yuval Rabani. Improved lower bounds for embeddings into  $l_1$ . *SIAM J. Comput.*, 38(6):2487–2498, 2009. doi:10.1137/060660126.

- 16 S. Kudekar, S. Kumar, M. Mondelli, H. D. Pfister, and R. Urbanke. Comparing the bit-map and block-map decoding thresholds of reed-muller codes on bms channels. In *2016 IEEE International Symposium on Information Theory (ISIT)*, pages 1755–1759, July 2016. doi:10.1109/ISIT.2016.7541600.
- 17 Ryan O’Donnell. *Analysis of boolean functions*. Cambridge University Press, 2014.
- 18 Ryan O’Donnell and Rocco A. Servedio. Learning monotone decision trees in polynomial time. *SIAM J. Comput.*, 37(3):827–844, 2007. doi:10.1137/060669309.
- 19 Michel Talagrand. On russo’s approximate zero-one law. *The Annals of Probability*, 22(3):1576–1587, 1994. URL: <http://www.jstor.org/stable/2245033>.

## A Missing Proofs

### A.1 Proof of Lemma 24

Towards the first property, we note that both functions  $T_{1-1/2d}f$  and  $T_{1-1/d}f$ , being averages of  $f$ , are bounded in the interval  $[0, 1]$ . Towards the second property, we note that  $I_i^1[h_d] \leq I_i^1[T_{1-1/2d}f] + I_i^1[T_{1-1/d}f]$  and that the latter are at most  $I_i^1[f]$ , again because  $T_{1-1/2d}f$  and  $T_{1-1/d}f$  are averages of  $f$ . Towards the third property, we note that by definition

$$\hat{h}_d(S) = \left( \left(1 - \frac{1}{2d}\right)^{|S|} - \left(1 - \frac{1}{d}\right)^{|S|} \right) \hat{f}(S).$$

The multiplicative factor in front of  $\hat{f}(S)$ , when  $d \leq |S| \leq 2d$ , is easily seen to be a constant. Towards the last property, we note that for each set  $S \neq \emptyset$ , its contribution to  $\text{var}(f)$  is  $\hat{f}(S)^2$  and to  $I[f]$  is  $|S|\hat{f}(S)^2$ , whereas the corresponding contributions to  $\sum_d \|h_d\|_2^2$  and  $\sum_d I[h_d]$  are similar up to the multiplicative factor

$$\sum_d \left( \left(1 - \frac{1}{2d}\right)^{|S|} - \left(1 - \frac{1}{d}\right)^{|S|} \right)^2.$$

It is enough to show that this sum is at most 1. Indeed, since each summand is square of a number in the range  $[0, 1]$ , we can ignore the squares and then it is just a telescoping sum upper bounded by 1.

### A.2 Proof of Lemma 25

In the following, sums run over all  $d$  that are powers of 2 unless the sum is restricted explicitly to a subset. Clearly,  $\sum_d W_{\approx d}[f] = \text{var}(f)$ , so it is enough to show that this sum over only those  $d \notin D_{\text{good}}$  is at most  $\frac{1}{2}\text{var}(f)$ . We consider two cases: those  $d$  that are “large”, that is  $d \geq T = \frac{4I^1[f]}{\text{var}(f)}$ , and those  $d$  that are “not large” but not in  $D_{\text{good}}$ . In the first case, we use Markov and in the second case, we note that there are only a few summands. Indeed, in the first case (using  $I[f] \leq I^1[f]$ ),

$$I[f] \geq T \cdot \sum_{d \geq T} W_{\approx d}[f], \quad \text{implying that} \quad \sum_{d \geq T} W_{\approx d}[f] \leq \frac{I[f]}{T} = \frac{I[f] \text{var}(f)}{4I^1[f]} \leq \frac{1}{4}\text{var}(f).$$

In the second case,  $d \leq T$ , so there are at most  $\log T$  chunks and when  $d \notin D_{\text{good}}$ , we have  $W_{\approx d}[f] \leq \frac{\text{var}(f)^2}{16 \cdot I^1[f]} = \frac{\text{var}(f)}{4T}$ . Hence

$$\sum_{d \leq T, d \notin D_{\text{good}}} W_{\approx d}[f] \leq \log T \cdot \frac{1}{4T} \cdot \text{var}(f) \leq \frac{1}{4}\text{var}(f).$$



# On Rich 2-to-1 Games

**Mark Braverman**

Department of Computer Science, Princeton University, NJ, USA  
mbraverm@cs.princeton.edu

**Subhash Khot**

Department of Computer Science, Courant Institute of Mathematical Sciences, New York University, NY, USA  
khot@cims.nyu.edu

**Dor Minzer**

Department of Mathematics, Massachusetts Institute of Technology, Cambridge, MA, USA  
minzer.dor@gmail.com

---

## Abstract

We propose a variant of the 2-to-1 Games Conjecture that we call the Rich 2-to-1 Games Conjecture and show that it is equivalent to the Unique Games Conjecture. We are motivated by two considerations. Firstly, in light of the recent proof of the 2-to-1 Games Conjecture [16, 6, 5, 17], we hope to understand how one might make further progress towards a proof of the Unique Games Conjecture. Secondly, the new variant along with perfect completeness in addition, might imply hardness of approximation results that necessarily require perfect completeness and (hence) are not implied by the Unique Games Conjecture.

**2012 ACM Subject Classification** Theory of computation → Problems, reductions and completeness

**Keywords and phrases** PCP, Unique-Games, Perfect Completeness

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.27

**Related Version** A full version of this paper is available at <https://eccc.weizmann.ac.il/report/2019/141/>.

**Funding** *Mark Braverman*: Research supported in part by the NSF Alan T. Waterman Award, Grant No. 1933331, a Packard Fellowship in Science and Engineering, and the Simons Collaboration on Algorithms and Geometry.

*Subhash Khot*: Supported by the NSF Award CCF-1422159, the Simons Collaboration on Algorithms and Geometry, and the Simons Investigator Award.

## 1 Introduction

The Unique Games Conjecture [11] is considered a central question in theoretical computer science. It has many applications to hardness of approximation (e.g. tight results for Max-Cut and Vertex Cover problems [14, 18]) and connections to algorithms, computational complexity, analysis, and geometry (e.g. see the surveys [22, 12, 13]). Recently, a related conjecture called the 2-to-1 Games Conjecture has been proved [16, 6, 5, 17]. This conjecture has many applications of its own, implies the Unique Games Conjecture “half-way” (in the technical sense, with “completeness”  $\frac{1}{2}$  instead of  $1 - o(1)$ ), and provides strong evidence in favor of the Unique Games Conjecture.

In light of this development, it is natural to ask whether the proof of the 2-to-1 Games Conjecture can somehow be extended to that of the Unique Games Conjecture. A straightforward extension does not look likely, so we raise the following possibility: perhaps the 2-to-1 Games Conjecture holds with additional structure on its instances, and hardness on such instances is then enough to prove the Unique Games Conjecture? In this paper, we investigate this possibility and make a concrete proposal in this regard. The proposal, that



© Mark Braverman, Subhash Khot, and Dor Minzer;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 27; pp. 27:1–27:20



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

we call the Rich 2-to-1 Games Conjecture, is described next along with the overall context. Our main result is that this variant of the 2-to-1 Games Conjecture turns out to be equivalent to the Unique Games Conjecture.

## 1.1 The Unique Games Conjecture

The Unique Games and 2-to-1 Games are specialized cases of the more general 2-Prover-1-Round Games.

► **Definition 1.** A 2P1R Games instance  $\Psi = (L \cup R, E, \Sigma_L, \Sigma_R, \Phi)$  consists of a regular, bipartite graph  $(L \cup R, E)$ , the alphabet  $\Sigma_L$  for the vertex set  $L$ , the alphabet  $\Sigma_R$  for the vertex set  $R$ , and a set of constraints  $\Phi = \{\phi_e\}_{e \in E}$ , one for each edge. Each vertex is supposed to receive a label from the respective alphabet. The constraint  $\phi_e$  for an edge  $e = (u, v) \in E, u \in L, v \in R$  is defined by a relation  $\phi_e \subseteq \Sigma_L \times \Sigma_R$ , thought of as the set of label-pairs to the vertices  $u$  and  $v$  that satisfy the constraint.

For  $1 \geq c > s > 0$ , and integers  $k, n$ , let  $\text{Gap-2P1R}_{k,n}[c, s]$  denote the promise problem where given a 2P1R Games instance  $\Psi$  as above with  $|\Sigma_L| = k, |\Sigma_R| = n$ , the problem is to distinguish whether there is a labeling to its vertices that satisfies  $c$  fraction of the constraints or whether every labeling satisfies at most  $s$  fraction of the constraints (we will often drop the subscripts  $k, n$  when clear from context).

2P1R Games are central to the theory of hardness of approximation and Probabilistically Checkable Proofs. These serve as canonical starting point for hardness reductions. The parameters of interest (from the viewpoint of making such reductions “work”) are: the alphabet size  $\max\{m, k\}$ , the “gap”  $(c, s)$ , and the nature of the relations  $\phi_e$ . Throughout this paper, the parameters  $k, n, c, s$  are thought of as constants and the size of the bipartite graph  $(L \cup R, E)$  as the instance size.

The 2P1R Games studied in applications are almost exclusively “Projection Games”, i.e. instances in which  $|\Sigma_L| \geq |\Sigma_R|$  and the constraint on each edge  $e = (u, v)$  is defined by a mapping  $\pi_e: \Sigma_L \rightarrow \Sigma_R$ ; the relation  $\phi_e$  is then  $\phi_e = \{(\sigma, \pi_e(\sigma)) \mid \sigma \in \Sigma_L\}$ , so that for every label to vertex  $u$ , there is a unique label to the vertex  $v$  that satisfies the constraint. We will restriction to Projection Games henceforth and denote the corresponding gap problem as **Gap-Projection**.

In the language of 2P1R Games, the celebrated PCP Theorem [8, 2, 1] states that  $\text{Gap-Projection}_{7,2}[1, s]$  is NP-hard for some absolute constant  $s < 1$ . Combining the PCP Theorem and Raz’s Parallel Repetition Theorem [20] gives the very important theorem that  $\text{Gap-Projection}_{k,n}[1, s]$  is NP-hard for every constant  $s > 0$  and with the alphabet size at most polynomial in  $\frac{1}{s}$ .

For an integer  $d$  (thought of as a small constant, say  $d = 2$ ), a  $d$ -to-1 Games instance is a Projection Games instance in which  $|\Sigma_L| = d \cdot |\Sigma_R|$  and the projection map  $\pi_e: \Sigma_L \rightarrow \Sigma_R$  defining the constraint is a  $d$ -to-1 map. The 1-to-1 Games are more commonly called the Unique Games and were studied by Feige and Lovasz [9] (in a different context). The corresponding gap versions are denoted as **Gap- $d$ -to-1** and **Gap-Unique** and the alphabet sizes are identified by one parameter  $n$  such that  $|\Sigma_L| = d \cdot n$  and  $|\Sigma_R| = n$ . The conjectures made in [11] are stated below (we take liberty to modify statements slightly regarding the issue of perfect versus imperfect completeness):

► **Conjecture 2 (Unique Games Conjecture).** For every constant  $\varepsilon > 0$ , there is a sufficiently large integer  $n$  such that  $\text{Gap-Unique}_n[1 - \varepsilon, \varepsilon]$  is NP-hard.

► **Conjecture 3** (*d*-to-1 Games Conjecture). *For every constant  $\varepsilon > 0$ , there is a sufficiently large integer  $n$  such that  $\text{Gap-d-to-1}_n[1 - \varepsilon, \varepsilon]$  is NP-hard.*

► **Conjecture 4** (*d*-to-1 Games Conjecture with Perfect Completeness). *For every constant  $\varepsilon > 0$ , there is a sufficiently large integer  $n$  such that  $\text{Gap-d-to-1}_n[1, \varepsilon]$  is NP-hard.*

In a recent development, the 2-to-1 Games Conjecture is proved in a sequence of papers [16, 6, 5, 17] (with additional contributions from [3, 15]), also proving as a simple corollary that  $\text{Gap-Unique}[\frac{1}{2}, \varepsilon]$  is NP-hard (for every  $\varepsilon > 0$  and for sufficiently large alphabet size). This gives a strong evidence towards correctness of the Unique Games Conjecture (which prior to this development was viewed skeptically by most researchers).

## 1.2 The Rich 2-to-1 Games

One naturally asks whether the proof of the 2-to-1 Games Conjecture extends, without substantial effort, to that of the Unique Games Conjecture. We do not believe this to be the case and instead make the following proposal and conjecture. We conjecture that the 2-to-1 Games Conjecture holds with additional structure on its instances (referred to as “richness”) and is then enough to prove the Unique Games Conjecture (in fact is equivalent to it). The new conjecture and the notion of richness are well-motivated as explained later on.

Let  $\Psi = (L \cup R, E, \Sigma_L, \Sigma_R, \Phi)$  be a 2-to-1-Game, with  $|\Sigma_L| = 2n$  and  $|\Sigma_R| = n$ . Fix a vertex  $u \in L$ . Let  $e = (u, v) \in E$  be an edge incident on  $u$  and let  $\pi_e$  be the 2-to-1 projection defining that constraint. The map defines a partition of  $\Sigma_L$  as  $\Sigma_L = \bigcup_{\rho \in \Sigma_R} \pi_e^{-1}(\rho)$  into disjoint sets of size 2. Let us denote by  $\mathcal{P}(u)$  the distribution over partitions of  $\Sigma_L$  into sets of size 2, given by first sampling a uniformly random edge  $e = (u, v)$  incident on  $u$  and then outputting the partition of  $\Sigma_L$  as above.

► **Definition 5.** *An instance of Rich 2-to-1 Games is an instance of 2-to-1 Games with the additional property that for every vertex  $u \in L$ , the distribution  $\mathcal{P}(u)$  is uniform over all partitions of  $\Sigma_L$  into sets of size 2.*

We now state the new conjecture (and also throw in a stronger version with perfect completeness). Our main result is that it is equivalent to the Unique Games Conjecture.

► **Conjecture 6** (Rich 2-to-1 Games Conjecture). *For every constant  $\varepsilon > 0$ , there is a sufficiently large integer  $n$  such that  $\text{Gap-Rich-2-to-1}_n[1 - \varepsilon, \varepsilon]$  is NP-hard.*

► **Conjecture 7** (Rich 2-to-1 Games Conjecture with Perfect Completeness). *For every constant  $\varepsilon > 0$ , there is a sufficiently large integer  $n$  such that  $\text{Gap-Rich-2-to-1}_n[1, \varepsilon]$  is NP-hard.*

► **Theorem 8** (Main Result). *The Unique Games Conjecture 2 and the Rich 2-to-1 Games Conjecture 6 are equivalent.*

The reduction from Unique Games to Rich 2-to-1 Games is straightforward, and is given in the full version of the paper. The reverse reduction requires new analytic results to analyze it. These results are stated in Section 3 and proved in Section 4. The reduction itself is presented in Section 5.

## 1.3 Motivation to Study the Rich 2-to-1 Games

We now explain how the notion of richness arises from natural (but admittedly technical) considerations. In short, the notion of richness is tailor-made so as to ensure the “sub-code covering” property; this property was identified and used in [19] and was crucial in the

proof of the 2-to-2 Games Conjecture [16, 6] (however there are differences that are outlined below). We then comment on how the notion of richness might be useful towards proving the Unique Games Conjecture and towards proving hardness of approximation results with perfect completeness. These comments are speculative in nature.

### Sub-code Covering Property

We describe, at a very high level, a typical PCP reduction starting with an instance of a Projection Game.<sup>1</sup> We admit that the description might not be friendly to a reader who is not already somewhat familiar with the area.

Let  $\Psi = (L \cup R, E, \Sigma_L, \Sigma_R, \Phi)$  be an instance of Projection Game. In the reduction (or equivalently the PCP proof), each vertex  $u \in L$  is replaced by a string  $\text{Enc}^*(u) \in [m]^{k_L}$  which is intended to be the encoding of the supposed label of  $u$  via an encoding scheme  $\text{Enc} : \Sigma_L \rightarrow [m]^{k_L}$ . The encoding scheme is chosen a priori. Here  $[m]$  is the proof alphabet (e.g.  $\{0, 1\}$ ) and  $k_L$  is the encoding length. Similarly, each vertex  $v \in R$  is replaced by a string  $\text{Enc}^*(v) \in [m]^{k_R}$  which is intended to be the encoding of the supposed label of  $v$  via the encoding scheme  $\text{Enc} : \Sigma_R \rightarrow [m]^{k_R}$ . For convenience, we use the same notation, namely  $\text{Enc}(\cdot)$ , to denote both encodings. Also, similar notation, namely  $\text{Enc}^*(\cdot)$  and  $\text{Enc}(\cdot)$ , is used to emphasize their relationship: the latter is a true encoding whereas the former is a purported encoding.

The task of the PCP verifier is to check, given a purported proof and an edge  $e = (u, v) \in E$ ,

- that the strings  $\text{Enc}^*(u)$  and  $\text{Enc}^*(v)$  in the purported proof are indeed codewords, i.e. that they are same as  $\text{Enc}(\sigma)$  for some label  $\sigma \in \Sigma_L$  and  $\text{Enc}(\rho)$  for some label  $\rho \in \Sigma_R$  respectively.
- that  $\pi_e(\sigma) = \rho$  where  $\pi_e : \Sigma_L \rightarrow \Sigma_R$  is the projection map defining the constraint.

These two tasks are referred to as the codeword test and the consistency test respectively and are often somehow incorporated into a single combined test (as seen below). Further, a combination of necessity and convenience dictates that:

- One needs to work with a relaxed conclusion that  $\text{Enc}^*(u)$  and  $\text{Enc}^*(v)$  are close to some codewords  $\text{Enc}(\sigma)$  and  $\text{Enc}(\rho)$  respectively so that  $\pi_e(\sigma) = \rho$ . This amounts to decoding or (more often) list-decoding the given strings  $\text{Enc}^*(u)$  and  $\text{Enc}^*(v)$ .
- One needs that the codeword  $\text{Enc}(\rho)$  is a “sub-code” of the codeword  $\text{Enc}(\sigma)$  whenever  $\pi_e(\sigma) = \rho$ . Specifically, for every location  $x \in [k_R]$  on the  $v$ -side, there is a location  $\pi_e^{-1}(x) \in [k_L]$  on the  $u$ -side such that  $\text{Enc}(\sigma)[\pi_e^{-1}(x)] = \text{Enc}(\rho)[x]$  whenever  $\pi_e(\sigma) = \rho$ .

Now we are ready to describe a typical PCP test. It picks  $v \in R$  randomly and generates query locations  $x_1, \dots, x_k$  for the codeword tester of the purported codeword  $\text{Enc}^*(v)$  along with a predicate  $P(\text{Enc}^*(v)[x_1], \dots, \text{Enc}^*(v)[x_k])$  that would determine whether the test accepts or rejects. However this test and the query locations are only virtual. To define the actual test and the query locations, one uses the property that the encoding  $\text{Enc}(\rho)$  of the supposed label  $\rho$  of  $v \in R$  is a sub-code of the encoding  $\text{Enc}(\sigma_i)$  of the supposed label  $\sigma_i$  of a neighbor  $u_i \in L$  of  $v$ ,  $e_i = (u_i, v)$ . Thus, one may read-off the symbol  $\text{Enc}^*(v)[x_i]$  from the corresponding symbol  $\text{Enc}^*(u_i)[y_i]$  for appropriate location  $y_i = \pi_{e_i}^{-1}(x_i)$  therein. More specifically, the test picks random, independent neighbors  $u_1, \dots, u_k \in L$  of  $v$ , and tests the predicate  $P(\text{Enc}^*(u_1)[y_1], \dots, \text{Enc}^*(u_k)[y_k])$ . This completes the description of a typical

<sup>1</sup> This paradigm is referred to as the “Inner/Outer PCP” in literature, but we avoid the usage of this terminology.



PCP test. To make this approach “work” however, more is needed. To see the difficulty involved, let’s assume that the (virtual) codeword test succeeds perfectly for every  $v \in R$ , i.e. that  $\text{Enc}^*(v) = \text{Enc}(\rho(v))$  for some label  $\rho(v)$  (that depends on  $v$ ). Looking at things from the perspective of some fixed  $u \in L$ , this amounts to saying that the purported encoding  $\text{Enc}^*(u)$  has, as its sub-strings, correct sub-codewords  $\text{Enc}(\rho(v_j))$  for all neighbors  $v_j \in R$  of  $u$ . Can we conclude now that  $\text{Enc}^*(u)$  is also a correct codeword or at least resembles a correct codeword? Not necessarily and that’s the trouble.

It is possible that the sub-codewords  $\text{Enc}(\rho(v_j))$  (or rather the set of their locations) constitute only a negligible portion of the purported codeword  $\text{Enc}^*(u)$  “on the larger side” (or rather the set of its locations). If so, the consistency of  $\text{Enc}^*(u)$  with all its correct sub-codewords would not say anything about correctness of  $\text{Enc}^*(u)$  itself. Clearly, the disparity in the encoding lengths  $k_L$  and  $k_R$  on the two sides and the number of neighbors  $v$  for a fixed  $u \in L$ , both have bearing on this issue. In [19], the authors defined the “sub-code covering property” that is informally stated as follows.

► **Definition 9 (Informal).** *The encoding scheme  $\text{Enc}(\cdot)$  along with the Projection Game structure is said to achieve sub-code covering property if for every fixed  $u \in L$ , the “pull-back distribution” on the (query) location  $y \in [k_L]$  as described next is statistically close to the uniform distribution over  $[k_L]$ . The pull-back distribution is defined by picking a random neighbor  $v \in R$  of  $u$ ,  $e = (u, v)$ , then picking a uniformly random location  $x \in [k_R]$  and letting  $y = \pi_e^{-1}(x)$ .*

In [19], the authors managed to achieve the sub-code covering property using Hadamard encoding (which sufficed for the application therein). This technique was subsequently used in the proof of the 2-to-1 Games Conjecture [16, 6] using Grassmann encoding (which again sufficed for the application therein). The Hadamard and Grassmann codes have length polynomial in the alphabet size  $|\Sigma_L|$  and  $|\Sigma_R|$  and while there is still a big disparity between the encoding lengths on the two sides, it is possible to arrange for a vertex  $u \in L$  to have sufficiently many neighbors  $v \in R$  and achieve the sub-code covering property (we omit the details). A serious restriction however is that using Hadamard and Grassmann encodings requires the projections  $\pi_e$  as well as the PCP test to be linear (limiting the efficacy of this approach).

## Long Code and Richness

In this paper, we attempt to work with the so-called Long encoding (defined below; this is extremely important in Unique Games based reductions). As is well-said, the Long code is too long. Its length is exponential in the alphabet size, making the disparity in encoding lengths on the two sides insurmountable (as far as we foresee). Still, we attempt to identify a scenario where the sub-code property is achievable using Long codes, possibly in a more relaxed sense. Indeed, we are able to do so when  $|\Sigma_L| = 2|\Sigma_R|$ , the projections  $\pi_e$  are 2-to-1, and the game is “rich” (meaning, for a fixed  $u \in L$ , for its random neighbor  $v \in R$ ,  $e = (u, v)$ , the partition of  $\Sigma_L$  into sets of size 2 induced by the projection  $\pi_e$  is uniform among all possible such partitions). We informally state this observation below.

► **Lemma 10 (Informal).** *The Long code along with the Rich 2-to-1 Game structure achieves a relaxed sub-code covering property in the following sense. For every fixed  $u \in L$ , the “pull-back distribution” on the (query) location  $y \in [k_L]$  as described in Definition 9 has the property that for most locations  $y \in [k_L]$ , their probability under the pull-back distribution is not much larger than their probability under uniform distribution on  $[k_L]$ .*

Formally, let  $\Sigma_L = [2n] = \{1, \dots, 2n\}$  and  $\Sigma_R = [n] = \{1, \dots, n\}$ . Fix a vertex  $u \in L$  in a Rich 2-to-1 Game and consider its randomly chosen neighbor  $v \in R$ . Then, by definition of richness,  $\pi = \pi_{(u,v)} : [2n] \rightarrow [n]$  is a uniformly random 2-to-1 map.

The  $m$ -ary Long code for the label of  $u$  corresponds to a function  $F : [m]^{2n} \rightarrow [m]$  and the codeword for the label  $i_0 \in [2n]$  corresponds to the  $i_0^{th}$  dictatorship function

$$\text{Dict}_{i_0}(z) = \text{Dict}_{i_0}(z_1, \dots, z_{2n}) = z_{i_0}.$$

Similarly, the Long code for the label of  $v$  corresponds to a function  $G : [m]^n \rightarrow [m]$  and the codeword for the label  $j_0 \in [n]$  corresponds to the  $j_0^{th}$  dictatorship function

$$\text{Dict}_{j_0}(x) = \text{Dict}_{j_0}(x_1, \dots, x_n) = x_{j_0}.$$

We observe that if one defines for  $x \in [m]^n$ ,  $\pi^{-1}(x) \in [m]^{2n}$  by letting  $\pi^{-1}(x)_i = x_{\pi(i)}$  for all  $i \in [2n]$ , it indeed holds that

$$\text{Dict}_{i_0}(\pi^{-1}(x)) = \text{Dict}_{j_0}(x) \quad \text{whenever } \pi(i_0) = j_0.$$

In this sense, the encoding corresponding to  $v$  is a sub-code of the encoding corresponding to  $u$ . A location  $z$  from the pull-back distribution on  $[m]^{2n}$  is sampled by first picking a uniformly random 2-to-1 map  $\pi : [2n] \rightarrow [n]$ , picking  $x \in [m]^n$  uniformly, and letting  $z = \pi^{-1}(x)$ . Clearly, this distribution is supported only on  $z \in [m]^{2n}$  for which each  $s \in [m]$  appears an even number of times as its co-ordinate, and hence is statistically far from the uniform distribution on  $[m]^{2n}$ . On the other hand, we show that for “typical”  $z \in [m]^{2n}$  (those for which all  $s \in [m]$  occur roughly equal number of times as its coordinate), its probability under the pull-back distribution is at most a constant times its probability under the uniform distribution. We refer the reader to Lemma 27 for a formal statement.

We have explained how the notion of richness is tailor-made to achieve the sub-code covering property for the Long code (albeit in a more relaxed sense). We now describe two motivations to study Rich 2-to-1 Games. Our comments are speculative, but we hope that these lead to fruitful research directions.

## Hardness Results with Perfect Completeness?

From the discussion so far, it is evident that Rich 2-to-1 Games could be an excellent problem to reduce from. In particular, we show that it can be reduced to the Unique Games problem, and is equivalent to the latter. In light of this equivalence, why not just stick to the Unique Games Conjecture then? The additional advantage of using the Rich 2-to-1 Games Conjecture could be that this conjecture could hold even with perfect completeness. This could be useful towards proving hardness of approximation results where perfect completeness is essential. We cite couple of plausible candidates where hardness results could follow from the Rich 2-to-1 Games Conjecture with perfect completeness:

- Hardness of coloring 3-colorable graphs with a constant number of colors.
- Hardness of CSPs (constraint satisfaction problems) on satisfiable instances. A concrete example is the query-efficient dictatorship test with perfect completeness that is proposed and analyzed in [21, 4]. Therein, one does not know how to translate the dictatorship test to a hardness result, lacking a suitable, conjectured hard problem to reduce from.

We remark that such results could follow by developing the appropriate analytic machinery on specialized domains (minor adjustments to the reduction are needed). A concrete example (related to the problem of proving hardness of coloring 3-colorable graphs with a

constant number of colors) is the multi-slice with an appropriate noise operator. Namely,  $V = \left\{ x \in \{0, 1, 2\}^{6n} \mid \text{the number of 0's, 1's and 2's in } x \text{ is } 2n \right\}$ , with the noise operator  $T$  that acts on  $V$  in the following way: given  $x$ , randomly change half of the 0-valued coordinates in  $x$  to 1's, and the rest into 2's, and similarly for the 1-valued and 2-valued coordinates. This operator can naturally be viewed as an averaging operator over functions, and one would need a “Majority is Stablest” type bound: if all of the low-degree influences of  $f: V \rightarrow \{0, 1\}$  are small, then  $\langle f, Tf \rangle$  is bounded away from 0.

More ambitiously, one could hope that by developing the necessary analytical tools on such non-classical domains, any dictatorship test with perfect completeness could be used to prove an NP-hardness result for the corresponding predicate, assuming Conjecture 7. We leave further investigation along this direction to future works.

## Making Games Richer?

One might argue that since 2-to-1 Games are now known to be hard, we should now work towards showing that “rich” 2-to-1 Games are hard as well, showing in turn that the Unique Games are hard. It might be possible to consider “degree of richness” and design a sequence of reductions that successively achieve higher degree of richness, finally achieving full richness as in the definition of Rich 2-to-1 Games.

Formally, let  $\mathcal{F}$  be a family of partitions of  $[2n]$  into sets of size 2 each. A 2-to-1 Game is called  $\mathcal{F}$ -rich if for every fixed vertex  $u \in L$ , for its random neighbor  $v \in R$ , the partition of  $[2n]$  induced by the projection  $\pi = \pi_{(u,v)}$  is uniform over the family  $\mathcal{F}$ . We defined the game to be rich if it is  $\mathcal{F}_{\text{all}}$ -rich, where  $\mathcal{F}_{\text{all}}$  is the family of all such partitions possible.

As is the case in the proof of the 2-to-1 Games Conjecture [16, 6], the 2-to-1 Games shown to be hard therein are  $\mathcal{F}_{\text{lin}}$ -rich. Here  $[2n]$  is identified with the additive group  $GF(2)^k$  and  $\mathcal{F}_{\text{lin}}$  consists of one partition for every  $b \in GF(2)^k, b \neq 0$  that induces the “linear pairing”  $(x, x + b)$  for all  $x \in GF(2)^k$ . We float the idea to define a sequence of families

$$\mathcal{F}_0 = \mathcal{F}_{\text{lin}} \subseteq \mathcal{F}_1 \dots \subseteq \mathcal{F}_T = \mathcal{F}_{\text{all}},$$

and design a sequence of reductions achieving  $\mathcal{F}_j$ -richness successively from  $j = 0$  (which we now know) to  $j = T$  (proving the Rich 2-to-1 Games Conjecture and hence the Unique Games Conjecture).

## 2 Preliminaries

**Notation:** We denote by  $[n]$  the set  $\{1, \dots, n\}$  and by  $[n]_d$  the set of ordered  $d$ -tuples of elements of  $[n]$  consisting of distinct elements. The set of all permutations on  $[n]$  is denoted by  $S_n$  and the set of all 2-to-1 mappings  $\pi: [2n] \rightarrow [n]$  is denoted by  $S_{2n,n}$ .

We consider functions  $f: [m]^n \rightarrow \mathbb{R}$ . The distribution on  $[m]^n$  is, by default, uniform (but we will have occasions to consider non-uniform distributions and if so, it will be clear from the context). A sample  $x \in [m]^n$  will, by default, denote a uniform sample. For  $p \geq 1$ , the  $p$ -norm is defined in the standard manner,  $\|f\|_p = \mathbb{E}_x [|f(x)|^p]^{1/p}$ . The inner product of two functions is  $\langle f, g \rangle = \mathbb{E}_x [f(x)g(x)]$ .

Throughout the paper,  $C(m), C(K, m), C(d, K, m)$  etc will denote a constant that depends on the respective parameters and this constant could change from time to time.

## 2.1 Basic Analytic Notions

We recall the standard way to express  $f: [m]^n \rightarrow \mathbb{R}$  in the Fourier basis. Here, it is more convenient to define  $[m] = \mathbb{Z}_m = \{0, 1, \dots, m-1\}$  with the additive group structure. Let  $\{Y_s : [m] \rightarrow \mathbb{R} \mid s \in [m]\}$  be an orthonormal set of random variables with  $Y_0 \equiv 1$ . One can then express a function  $f: [m]^n \rightarrow \mathbb{R}$  uniquely as a “multi-linear” polynomial in random variables  $\{X_{i,s} \mid 1 \leq i \leq n, s \in [m]\}$  where for each  $1 \leq i \leq n$ , the  $\{X_{i,s}\}$  are copies of  $\{Y_s\}$ , and are independent for different  $i$ . A degree- $d$  “monomial” looks like  $\prod_{j=1}^d X_{i_j, s_j}$  with  $s_j \neq 0$  and  $i_j$  distinct for  $1 \leq j \leq d$ . The degree of a polynomial is the maximum degree of its non-zero monomials.

► **Theorem 11 (Hypercontractivity).** *Let  $f: [m]^n \rightarrow \mathbb{R}$  be a function of degree at most  $d$ . Then for all  $p \geq 2$ ,  $\|f\|_p \leq \sqrt{m(p-1)}^d \|f\|_2$ .*

► **Definition 12.** *The noise operator  $T_{1-\varepsilon}$  acts on functions  $f: [m]^n \rightarrow \mathbb{R}$  by defining  $T_{1-\varepsilon}f(x) = \mathbb{E}_{z \sim_{1-\varepsilon} x} [f(z)]$ . Here  $z \sim_{1-\varepsilon} x$  denotes a random input  $z$  that is  $(1-\varepsilon)$ -correlated with  $x$ , i.e. independently for each coordinate  $1 \leq i \leq n$ , the  $i^{\text{th}}$  coordinate of  $z$  equals the  $i^{\text{th}}$  coordinate of  $x$  with probability  $1-\varepsilon$  and is sampled uniformly from  $[m]$  with probability  $\varepsilon$ .*

► **Definition 13.** *The influence of a coordinate  $i \in [n]$  on a function  $f: [m]^n \rightarrow \mathbb{R}$  is defined by  $I_i[f] = \mathbb{E}_x [(f(x) - \mathbb{E}_{s \in [m]} [f(x + se_i)])^2]$ .*

► **Lemma 14.** *Let  $f: [m]^n \rightarrow \mathbb{R}$  be a function and  $i \in [n]$ .*

$$\frac{1}{4} \mathbb{E}_{x, s \in [m]} [(f(x) - f(x + se_i))^2] \leq I_i[f] \leq \mathbb{E}_{x, s \in [m]} [(f(x) - f(x + se_i))^2].$$

**Proof.** Deferred to the full version. ◀

► **Definition 15.** *Let  $f^{\leq d}$  and  $f^{> d}$  denote the parts of  $f$  with degree at most  $d$  and larger than  $d$  respectively. The degree- $d$  influence of a variable  $i \in [n]$  on  $f$  is defined as  $I_i^{\leq d}[f] = I_i[f^{\leq d}]$ .*

We need the following noise-stability result of [7]. It upper-bounds the noise-stability of functions all of whose influences are low.

► **Theorem 16.** *For every integer  $m \geq 2$  and constants  $\varepsilon, \theta > 0$ , there is a sufficiently small constant  $\delta > 0$ , such that the following holds. Let  $f: [m]^n \rightarrow [0, 1]$  with  $\mathbb{E}[f] \leq \theta$  and assume that for all  $i \in [n]$ ,  $I_i[f] \leq \delta$ . Then  $\langle f, T_{1-\varepsilon}f \rangle \leq 2\Gamma_{1-\varepsilon}(\theta)$ .*

The function  $\Gamma_{1-\varepsilon}(\theta)$  is defined in [14] and the only property we need is that for a fixed  $\varepsilon > 0$ ,  $\frac{\Gamma_{1-\varepsilon}(\theta)}{\theta} \rightarrow 0$  as  $\theta \rightarrow 0$ . A known upper bound is  $\Gamma_{1-\varepsilon}(\theta) \leq C(\varepsilon) \theta^{2/(2-\varepsilon)}$ .

**Functions with range  $[m]$ :** We also consider functions  $F: [m]^n \rightarrow [m]$ , which are more convenient to view as  $F: [m]^n \rightarrow \Delta_m$  where  $\Delta_m$  is the standard  $m$ -dimensional simplex,  $\Delta_m = \{(t_0, \dots, t_{m-1}) \mid \forall i, t_i \geq 0, \sum_{i=0}^{m-1} t_i = 1\}$ . The value  $s \in [m]$  is then identified with the vertex  $e_s \in \Delta_m$  of the simplex. Usually, we consider the function  $F: [m]^n \rightarrow \Delta_m$  as a vector of  $[0, 1]$ -valued functions  $(F_0, F_1, \dots, F_{m-1})$ .

---

<sup>2</sup>  $e_i$  denotes an input that is 1 in the  $i^{\text{th}}$  coordinate and zero otherwise.

## 2.2 Hypercontractivity on the Symmetric Group and the 2-to-1 Mappings Domain

In this section, we give the basic background towards analyzing functions on the symmetric group and state the hypercontractive result we need. We consider functions  $F: S_n \rightarrow \mathbb{R}$ . For  $S, T \in [n]_k$ ,  $S = (i_1, \dots, i_k)$ ,  $T = (j_1, \dots, j_k)$ , we write  $\pi(S) = T$  if  $\pi(i_1) = j_1, \dots, \pi(i_k) = j_k$ . Let  $1_{\pi(S)=T}$  be the indicator function on  $S_n$  indicating that  $\pi(S) = T$ .

► **Definition 17.** For  $d = 0, \dots, n$ , let  $V_d(S_n)$  be the linear subspace spanned by all functions  $\{1_{\pi(S)=T} \mid S, T \in [n]_k, 0 \leq k \leq d\}$ . We say the “degree” of  $F$  is (at most)  $d$  if  $F \in V_d(S_n)$ .

► **Definition 18.** A degree- $d$  function  $F: S_n \rightarrow \mathbb{R}$  is called  $\varepsilon$ -pseudo-random if for any  $S, T \in [n]_d$ ,  $\mathbb{E}_{\pi: \pi(S)=T} [F[\pi]^2] \leq \varepsilon$ .

We need the following hypercontractive inequality from [10]. We will use it to show certain concentration properties of functions on the symmetric group (or more precisely on the 2-to-1 mappings domain defined next).

► **Theorem 19.** Let  $F: S_n \rightarrow \mathbb{R}$  be a degree- $d$ ,  $\varepsilon$ -pseudo-random function. Then  $(C(d) = 2^{40d^2} \text{ suffices}) \mathbb{E}_{\pi} [F[\pi]^4] \leq C(d)\varepsilon^2$ .

What we really need to analyze are functions on the 2-to-1 mappings domain  $S_{2n,n}$ , i.e. the set of 2-to-1 mappings  $\pi: [2n] \rightarrow [n]$ . We define the notion of degree of a function  $F: S_{2n,n} \rightarrow \mathbb{R}$  in a similar manner. For  $S \in [2n]_{2k}$ ,  $T \in [n]_k$ ,  $S = (i_1, i'_1, \dots, i_k, i'_k)$ ,  $T = (j_1, \dots, j_k)$ , we write  $\pi(S) = T$  if  $\pi(i_1) = \pi(i'_1) = j_1, \dots, \pi(i_k) = \pi(i'_k) = j_k$ . Let  $1_{\pi(S)=T}$  be the indicator function on  $S_{2n,n}$  indicating that  $\pi(S) = T$ .

► **Definition 20.** For  $d = 0, \dots, n$ , let  $V_d(S_n)$  be the linear subspace spanned by all functions  $\{1_{\pi(S)=T} \mid S \in [2n]_{2k}, T \in [n]_k, 0 \leq k \leq d\}$ . We say the “degree” of  $F$  is (at most)  $d$  if  $F \in V_d(S_n)$ .

► **Definition 21.** A degree- $d$  function  $F: S_{2n,n} \rightarrow \mathbb{R}$  is called  $\varepsilon$ -pseudo-random if for any  $S \in [2n]_{4d}$  and  $T \in [n]_{2d}$  we have  $\mathbb{E}_{\pi: \pi(S)=T} [F[\pi]^2] \leq \varepsilon$ .

► **Theorem 22.** Let  $F: S_{2n,n} \rightarrow \mathbb{R}$  be a degree- $d$ ,  $\varepsilon$ -pseudo-random function. Then  $(C(d) = 2^{160d^2} \text{ suffices}) \mathbb{E}_{\pi} [F[\pi]^4] \leq C(d)\varepsilon^2$ .

**Proof.** The proof proceeds by embedding  $S_{2n}$  into  $S_{2n,n}$ , and is deferred to the full version. ◀

## 3 Main Analytic Lemma

We now state our main analytic lemma. Let  $\pi \in S_{2n,n}$  be a 2-to-1 map. We recall that for  $x \in [m]^n$ , its “pull-back”  $\pi^{-1}(x) \in [m]^{2n}$  is defined as  $\pi^{-1}(x)_i = x_{\pi(i)}$  for  $i \in [2n]$ . For a function  $f: [m]^{2n} \rightarrow \mathbb{R}$ , the “restriction”  $f|_{\pi}: [m]^n \rightarrow \mathbb{R}$  is defined as <sup>3</sup>  $f|_{\pi}(x) = f(\pi^{-1}(x))$ . Our main lemma states, loosely speaking, that if  $f$  is a low-degree, bounded function and if  $\pi \in S_{2n,n}$  is a random 2-to-1 map, then the influential co-ordinates of  $f$  and those of the restricted function  $f|_{\pi}$  are related. More specifically, it is unlikely to happen that  $f|_{\pi}$  has some influential co-ordinate  $j$  without either of  $i, i' \in \pi^{-1}(j)$  being influential for  $f$ .

<sup>3</sup> This is indeed restriction of  $f$  to the pull-back domain  $\pi^{-1}([m]^n)$ .

► **Lemma 23.** Fix the alphabet size  $m \geq 2$ . For every constants  $\delta, \zeta > 0$  and integer  $d \geq 1$ , there are sufficiently small constants  $\gamma = \gamma(m, \delta, \zeta), \tau = \tau(d, m, \delta, \zeta) > 0$  such that the following holds. Suppose  $f: [m]^{2n} \rightarrow [0, 1]$  is a function such that  $\|f^{>d}\|_2^2 \leq \gamma$ . Then

$$\Pr_{\pi} \left[ \exists j \in [n] : I_j[f|_{\pi}] \geq \delta \quad \wedge \quad \max_{i \in \pi^{-1}(j)} I_i^{\leq d}[f] \leq \tau \right] \leq \zeta.$$

For convenience, we prove a very similar lemma stated below, which considers the special case when  $f$  itself has no degree- $d$  influential variables at all. Its proof contains all the main ingredients and the above Lemma 23 then follows with minor modifications. Due to space constraints, we defer this derivation to the full version of the paper.

► **Lemma 24.** Fix the alphabet size  $m \geq 2$ . For every constants  $\delta, \zeta > 0$  and integer  $d \geq 1$ , there are sufficiently small constants  $\gamma = \gamma(m, \delta, \zeta), \tau = \tau(d, m, \delta, \zeta) > 0$  such that the following holds. Suppose  $f: [m]^{2n} \rightarrow [0, 1]$  is a function such that  $\|f^{>d}\|_2^2 \leq \gamma$  and moreover that for all  $i \in [2n]$ ,  $I_i^{\leq d}[f] \leq \tau$ . Then,  $\Pr_{\pi} [\exists j \in [n] : I_j[f|_{\pi}] \geq \delta] \leq \zeta$ .

► **Remark 25.** It is important that in the statements of the lemmas above,  $\gamma$  does not depend on  $d$ . When we apply these lemmas,  $f$  itself will be a smoothed version  $T_{1-\varepsilon}h$  for some  $[0, 1]$ -valued function  $h$ . Thus  $\|f^{>d}\|_2^2 \leq \gamma = 2^{-\Omega(d/\varepsilon)}$  and in fact  $d$  will be chosen sufficiently large so as to make  $\gamma$  sufficiently small (so the dependence “in practice” is really the other way round).

## 4 Proof of the Main Analytic Lemma

In this section, we prove Lemma 24 (and the proof of Lemma 23 follows by minor modifications). We will work, for the large part, with function  $g$  that is, roughly speaking,  $f^{\leq d}$ . However, for technical reasons, we will zero-out its values on a small set of “atypical” inputs that are outside a certain set  $E \subseteq [m]^{2n}$ . Formally,  $g = f^{\leq d} 1_E$  where  $1_E$  is the indicator of set  $E$ . Towards the end of the proof, we will relate influences of  $f$  and  $g$ . Motivation and overview of successive steps in the proof is presented as we go along.

### 4.1 The Pull-back Distribution

While trying to relate influences of a function  $g: [m]^{2n} \rightarrow \mathbb{R}$  to those of its restrictions  $g|_{\pi}$ , a technical hurdle is that the “pull-back distribution” on  $[m]^{2n}$  that we define next differs from the uniform distribution on  $[m]^{2n}$ . The pull-back distribution arises while considering the average of influences of  $g|_{\pi}$  over the choice of  $\pi$  whereas the influences of  $g$  itself are defined with respect to the uniform distribution. We are able to show that the pull-back distribution resembles the uniform distribution on  $[m]^{2n}$  in a loose, but controlled manner.

► **Definition 26.** The pull-back distribution  $\nu_{2n,m}$  over  $[m]^{2n}$  is defined by the following process: sample  $\pi \in S_{2n,n}$ ,  $x \in [m]^n$  and output  $z = \pi^{-1}(x)$ .

Clearly, this distribution is supported only on  $z \in [m]^{2n}$  for which each  $s \in [m]$  appears an even number of times as its coordinate, and hence is statistically far from the uniform distribution on  $[m]^{2n}$ . On the other hand, we show that for “typical”  $z \in [m]^{2n}$ , its probability under the distribution  $\nu_{2n,m}$  is at most a constant times its probability under the uniform distribution (this and an additional related fact is all we need).

► **Lemma 27.** A point  $z \in [m]^{2n}$  is called  $K$ -roughly balanced if every value  $s \in [m]$  appears in  $\frac{2n}{m} \pm \sqrt{K \log m \frac{n}{m}}$  of the coordinates of  $z$ . For a  $K$ -roughly balanced point  $z$ ,

$$\nu_{2n,m}(z) \leq C(K, m) m^{-2n}.$$

**Proof.** Let  $A_s$  be the set of coordinates of  $z$  that are equal to  $s \in [m]$ , let  $a_s = |A_s|$ , and let  $a_s = \frac{2n+v_s}{m}$ . We may assume that all sets  $A_s$  are even-sized, since otherwise  $\nu_{n,m}(z) = 0$ . In this case,  $\nu_{n,m}(z)$  is equal to  $m^{-n}$  times the probability that for a random  $\pi \in S_{2n,n}$ ,  $z$  happens to be in the range of  $\pi^{-1}$ , or equivalently that  $\pi$  matches off each set  $A_s$  within itself. By Lemma 40, this probability is  $\frac{\binom{n}{a_0/2, \dots, a_{m-1}/2}}{\binom{2n}{a_0, \dots, a_{m-1}}}$ . Since  $z$  is  $K$ -roughly balanced, we have that  $|v_s| \leq \sqrt{(K \log m) m n}$ . Using Lemma 39, the ratio between the two multinomial coefficients is at most  $C(m) 2^{K \log m \cdot m} m^{-n} = C(K, m) m^{-n}$ . ◀

► **Remark 28.** We will often use the lemma above with additional conditioning on the choice of  $\pi$ , say for example that  $\pi$  is sampled uniformly with the condition  $\pi(2n-1) = \pi(2n) = n$ . The lemma continues to hold. The distribution  $\tilde{\nu}_{2n,m}$  on inputs  $z \in [m]^{2n}$  is now supported on  $z$  where every  $s \in [m]$  occurs an even number of times as its coordinate and moreover that  $z_{2n-1} = z_{2n}$ . Writing  $z = (\tilde{z}, z_{2n-1}, z_{2n})$ , if  $z$  is  $K$ -roughly balanced, then  $\tilde{z} \in [m]^{2n-2}$  is  $(K+1)$ -roughly balanced and the probability that  $\tilde{z}$  is output is  $C(K, m)$  times its probability under uniform distribution.

The lemma above immediately implies the following. It is then used to relate influences of  $g : [m]^{2n} \rightarrow \mathbb{R}$  to those of  $g|_\pi$  (the latter in expectation).

► **Lemma 29.** *Let  $h : [m]^{2n} \rightarrow [0, \infty)$  be a function supported only on  $K$ -roughly balanced inputs. Then  $\mathbb{E}_{z \sim \nu_{2n,m}} [h(z)] \leq C(K, m) \mathbb{E}_{z \in_R [m]^{2n}} [h(z)]$ .*

We now show how this is useful. Let  $g : [m]^{2n} \rightarrow \mathbb{R}$  and consider a random choice of  $\pi \in S_{2n,n}$  such that  $\pi(2n-1) = \pi(2n) = n$ . Such  $\pi$  can be chosen at random by first choosing  $\pi' \in S_{2(n-1), (n-1)}$  at random, letting  $\pi = \pi'$  on  $[2(n-1)]$ , and then extending by letting  $\pi(2n-1) = \pi(2n) = n$ . We wish to consider the expected influence of the  $n^{\text{th}}$  coordinate on the restriction  $g|_\pi$ .

► **Remark 30.** Here we specifically consider the  $n^{\text{th}}$  coordinate of  $g|_\pi$  under the requirement  $\pi(2n-1) = \pi(2n) = n$ . This is for notational convenience only and is without loss of generality. The same results hold for any given  $j^{\text{th}}$  coordinate of  $g|_\pi$  under the requirement that  $\pi(i) = \pi(i') = j$  for any given  $i \neq i' \in [2n]$ .

► **Lemma 31.** *Let  $g : [m]^{2n} \rightarrow \mathbb{R}$  be a function supported only on  $K$ -roughly balanced inputs. Then  $\mathbb{E}_\pi [I_n[g|_\pi]] \leq C(K, m) (I_{2n-1}[g] + I_{2n}[g])$ .*

**Proof.** Let  $e_{2n}$  be the input with the  $(2n)^{\text{th}}$  coordinate 1 and all other coordinates zero. Let  $e_{2n-1}$  be similarly defined and let  $e = e_{2n-1} + e_{2n}$ . By Lemma 14,

$$\mathbb{E}_\pi [I_n[g|_\pi]] \leq \mathbb{E}_{\substack{\pi, x \in [m]^n \\ s \in [m]}} \left[ \left( g(\pi^{-1}(x)) - g(\pi^{-1}(x) + s e) \right)^2 \right].$$

Let  $z = \pi^{-1}(x)$  so that  $z$  is distributed according to the distribution  $\tilde{\nu}_{2n,m}$  (see Remark 28). Since  $g$  is supported only on  $K$ -roughly balanced inputs, the term above is non-zero only if  $z$  is  $(K+1)$ -roughly balanced. Hence by Lemma 29, the above expectation is at most

$$C(K, m) \cdot \mathbb{E}_{z \in_R [m]^{2n}, s} \left[ \left( g(z) - g(z + s e) \right)^2 \right].$$

Note that we think of  $z \in_R [m]^{2n}$  as uniformly distributed now onwards. Using  $(a-b)^2 \leq 2(a-c)^2 + 2(c-b)^2$ , the last expectation is at most twice

$$\mathbb{E}_{z,s} \left[ \left( g(z) - g(z + s e_{2n-1}) \right)^2 \right] + \mathbb{E}_{z,s} \left[ \left( g(z + s e_{2n-1}) - g(z + s e_{2n-1} + s e_{2n}) \right)^2 \right].$$

Since the distribution of  $z \in [m]^{2n}$  is uniform, so is the distribution of  $z + s e_{2n-1}$  and hence these expectations are equal (up to a factor 4) to  $I_{2n-1}[g]$  and  $I_{2n}[g]$  respectively. ◀



## 4.2 The Function $G$ on $S_{2n,n}$ and its Pseudo-randomness

We seek to show that under appropriate conditions, if a function  $g : [m]^{2n} \rightarrow \mathbb{R}$  has all influences low, then with high probability over the choice of  $\pi$ , the same is true for the restriction  $g|_\pi$ . We begin by a (somewhat imprecise) proof-sketch.

Suppose that  $g$  has all influences low, say at most  $\tau$ . By above Lemma 31, the expected value of the influence  $I_n[g|_\pi]$ , over the choice of  $\pi$ , is at most  $O(\tau)$ . We would like to show that in fact  $I_n[g|_\pi]$  is at most  $O(\tau)$  with high probability over the choice of  $\pi$ . We would then argue that the same holds for the influence  $I_j[g|_\pi]$  for every  $1 \leq j \leq n$  (since consideration of the  $n^{\text{th}}$  coordinate was just for notational convenience), then take a union bound over all  $1 \leq j \leq n$ , and conclude that the restriction  $g|_\pi$  has all influences low.

However, such an argument requires strong probabilistic guarantees. It is natural to seek an upper bound on the higher moments of the random variable  $G[\pi] = I_n[g|_\pi]$ . We are able to do this, but only in a rather convoluted manner. We show that  $G[\pi]$  is pseudo-random as a function on  $S_{2n,n}$  (or strictly speaking, on  $S_{2(n-1),n-1}$  since  $\pi(2n-1) = \pi(2n) = n$  is pre-defined) in the sense of Definition 21. Concretely, we show that for small  $d$  and any sets  $|A| = 2(d-1)$ ,  $|B| = d-1$ , the conditional second moment  $\mathbb{E}_{\pi(A)=B} [G[\pi]^2]$  remains bounded by  $O(1)$  times (the unconditional second moment)  $\mathbb{E} [G[\pi]^2]$ . For notational convenience (only), one can think of

$$A = \{2(n-(d-1))-1, 2(n-(d-1)), \dots, 2(n-1)-1, 2(n-1)\}, \quad B = \{n-(d-1), \dots, n-1\},$$

and the event  $\pi(A) = B$  denotes the event that  $\pi(2(n-j)-1) = \pi(2(n-j)) = n-j$  for  $1 \leq j \leq d-1$  (and in addition,  $\pi(2n-1) = \pi(2n) = n$  is pre-defined, corresponding to  $j=0$ ).

This pseudo-randomness property then implies that the fourth moment  $\mathbb{E} [G[\pi]^4]$  is upper bounded by  $O(1)$  times (the square of the second moment)  $\mathbb{E} [G[\pi]^2]^2$ . This gives sufficiently strong guarantees to make the “with high probability” and union bound arguments to go through. Towards implementing the details of this proof, we need the following ad hoc sounding lemma. We then show how to use it and prove the desired pseudo-randomness property.

- **Lemma 32.** *Let a pair of inputs  $z_1, z_2 \in [m]^{2n}$  be chosen by two different methods:*
- *Choose a random  $\pi \in S_{2n,n}$ , then choose  $x_1, x_2 \in [m]^n$  at random, and then define  $z_i = \pi^{-1}(x_i)$ . Let  $\mu(z_1, z_2)$  denote the probability that the pair  $(z_1, z_2)$  is output.*
  - *Let  $A = \{2(n-(d-1))-1, 2(n-(d-1)), \dots, 2n-1, 2n\}$ ,  $B = \{n-(d-1), \dots, n\}$ , and the event  $\pi(A) = B$  denotes the event that  $\pi(2(n-j)-1) = \pi(2(n-j)) = n-j$  for  $0 \leq j \leq d-1$ . Let  $\mu_{\text{cond}}(z_1, z_2)$  denote the probability that the pair  $(z_1, z_2)$  is output by the method above, but conditional on the event  $\pi(A) = B$ .*

*Then if the pair  $(z_1, z_2)$  is “typical”, we have  $\mu_{\text{cond}}(z_1, z_2) \leq C(d, m) \mu(z_1, z_2)$ , where the pair  $(z_1, z_2)$  is “typical” if among the multi-set  $\{(z_1(i), z_2(i)) | 1 \leq i \leq 2n\}$  of their coordinates, each of the  $m^2$  patterns in  $[m] \times [m]$  appears at least  $\frac{2n}{20m^2}$  times.*

**Proof.** Among the multi-set  $\{(z_1(i), z_2(i)) | 1 \leq i \leq 2n\}$ , let the number of occurrences of the  $m^2$  possible patterns be  $v_1, \dots, v_{m^2}$ . We may assume that these numbers are all even since otherwise the pair  $(z_1, z_2)$  will never be output. The probability  $\mu(z_1, z_2)$  is equal to (using Lemma 40)  $\frac{\binom{\frac{v_1}{2}, \dots, \frac{v_{m^2}}{2}}{2n}}{\binom{v_1, \dots, v_{m^2}}{2n}}$ . Denote by  $u_1, \dots, u_{m^2}$  the number of occurrences of these patterns that appear in the  $2d$  coordinates of  $A$  so that  $2d = u_1 + \dots + u_{m^2}$ . The probability  $\mu_{\text{cond}}(z_1, z_2)$  is equal to (using Lemma 40 again)  $\frac{\binom{\frac{v_1-u_1}{2}, \dots, \frac{v_{m^2}-u_{m^2}}{2}}{2n-2d}}{\binom{v_1-u_1, \dots, v_{m^2}-u_{m^2}}{2n-2d}}$ . Applying Lemma 38,

we see that the numerator of the second fraction is at most  $C(d, m)$  times the numerator of the first fraction, and its denominator is at least  $c(d, m)$  times the denominator of the first fraction for some  $c(d, m) > 0$ , and hence we conclude that  $\mu_{\text{cond}}(z_1, z_2) \leq C(d, m)\mu(z_1, z_2)$  for a typical pair  $(z_1, z_2)$ .  $\blacktriangleleft$

► **Lemma 33.** *Let  $g : [m]^{2n} \rightarrow \mathbb{R}$  be a function supported on  $K$ -roughly balanced inputs. Let  $A = \{2(n - (d - 1)) - 1, 2(n - (d - 1)), \dots, 2(n - 1) - 1, 2(n - 1)\}$ ,  $B = \{n - (d - 1), \dots, n - 1\}$ . Then the random variable  $G[\pi] = I_n[g|\pi]$  satisfies  $(\pi(2n - 1) = \pi(2n) = n)$  is pre-defined)*

$$\mathbb{E}_{\pi(A)=B} [G[\pi]^2] \leq C(d, m) \mathbb{E} [G[\pi]^2] + 2^{-\Omega(\frac{n}{m^2})} \cdot C(d, K, m) \cdot \|g\|_4^4.$$

**Proof.** Denote  $h(z) = (g(z) - \mathbb{E}_{s \in [m]} [g(z + s e)])^2$  where the last two coordinates of  $e$  equal 1 and the rest are zero. By definition,  $\mathbb{E}_{\pi(A)=B} [G[\pi]^2]$  equals

$$\mathbb{E}_{\pi(A)=B} [I_n[g|\pi]^2] = \mathbb{E}_{\pi(A)=B} \left[ \mathbb{E}_{x_1, x_2} [h(\pi^{-1}(x_1))h(\pi^{-1}(x_2))] \right] = \mathbb{E}_{(z_1, z_2) \sim \mu_{\text{cond}}} [h(z_1)h(z_2)]. \quad (1)$$

We wish to upper bound this expression in terms of  $\mathbb{E} [G[\pi]^2]$  which may be written similarly as  $\mathbb{E} [G[\pi]^2] = \mathbb{E}_{(z_1, z_2) \sim \mu} [h(z_1)h(z_2)]$ , the two expectations being similar, but under different distributions  $\mu_{\text{cond}}$  and  $\mu$  respectively. The proof proceeds by splitting the expectation in (1) into two parts, over the pairs  $(z_1, z_2)$  that are typical versus that are atypical. For the first part, we upper bound using the above Lemma 32 and hence are able to “switch” to the distribution  $\mu$ . We now show how to upper bound the second part; this is by using Cauchy-Schwartz carefully and noting that only a negligible number of pairs are atypical. Let  $1_{\text{Bad}}$  denote the indicator of the event that the pair  $(z_1, z_2)$  is atypical. We note that the probability of this event is at most  $2^{-\Omega(\frac{n}{m^2})}$ . We wish to upper bound

$$\mathbb{E}_{(z_1, z_2) \sim \mu_{\text{cond}}} [h(z_1)h(z_2)1_{\text{Bad}}(z_1, z_2)] = \mathbb{E}_{(z_1, z_2) \sim \mu_{\text{cond}}} [h(z_1)1_{\text{Bad}}(z_1, z_2) \cdot h(z_2)1_{\text{Bad}}(z_1, z_2)].$$

By Cauchy-Schwartz, this is upper bounded by

$$\mathbb{E}_{(z_1, z_2) \sim \mu_{\text{cond}}} [h(z_1)^2 1_{\text{Bad}}(z_1, z_2)]. \quad (2)$$

Since  $g(z_1)$  is non-zero only on  $K$ -roughly balanced inputs  $z_1$ , the same holds for  $h(z_1)$  (possibly replacing  $K$  by  $K + 1$ ; we ignore this minor point). We may thus assume that  $z_1$  is  $K$ -roughly balanced. Provided that  $z_1$  is  $K$ -roughly balanced, the probability that  $(z_1, z_2)$  is atypical remains  $2^{-\Omega(\frac{n}{m^2})}$ . We note in addition that  $h(z_1)^2 \leq C(m) \cdot \mathbb{E}_{s \in [m]} [g(z_1 + s e)^4]$ , and that since  $z_1$  is  $K$ -roughly balanced, its probability under  $\mu_{\text{cond}}$  is at most  $C(d, K, m)$  times that under the uniform distribution on  $[m]^{2n}$  (by Lemma 27; the conditioning  $\pi(A) = B$  may give additional factor of  $m^d$ ). Putting these observations together, we upper bound (2), as desired, by  $2^{-\Omega(\frac{n}{m^2})} \cdot C(d, K, m) \cdot \mathbb{E}_{z \in [m]^{2n}} [g(z)^4]$ .  $\blacktriangleleft$

### 4.3 Using Hypercontractivity on $S_{2n, n}$

We now present the key hypercontractive argument, almost completing the proof as far as the function  $g = f^{\leq d} 1_E$  is concerned. In subsequent sections, we carry out the final steps relating influences of  $f$  and  $g$ .

► **Lemma 34.** *Let  $f : [m]^{2n} \rightarrow [0, 1]$  be a bounded function and  $E \subseteq [m]^{2n}$  be the set of  $K$ -roughly balanced inputs. Define  $g = f^{\leq d} \cdot 1_E$ , i.e.  $g$  is the low-degree part of  $f$ , but in addition zeroed out on the imbalanced inputs. Then*

$$\Pr_{\pi} [I_n[g|\pi] \geq \delta] \leq \frac{C(d, K, m)}{\delta^4} (I_{2n-1}[g]^4 + I_{2n}[g]^4).$$

**Proof.** We use Lemma 33 to conclude that

$$\mathbb{E}_{\pi(A)=B} [G[\pi]^2] \leq C(d, m) \mathbb{E} [G[\pi]^2] + 2^{-\Omega(\frac{n}{m^2})} \cdot C(d, K, m) \cdot \|g\|_4^4.$$

Towards bounding the second term, we observe

$$\|g\|_4^4 = \|f^{\leq d} 1_E\|_4^4 \leq \|f^{\leq d}\|_4^4 \leq (3m)^{2d} \|f^{\leq d}\|_2^4 \leq (3m)^{2d} \|f\|_2^4 \leq (3m)^{2d}.$$

Here we used the fact that  $f$  is a bounded function and Theorem 11. The  $2^{-\Omega(\frac{n}{m^2})}$  factor in the second term makes the term negligible. This term does not really affect subsequent arguments, so for the clarity of presentation, we take the liberty to ignore it henceforth. Thus we have  $\mathbb{E}_{\pi(A)=B} [G[\pi]^2] \leq C(d, m) \mathbb{E} [G[\pi]^2]$ . Since this holds for any  $|A| = 2d - 2, |B| = d - 1$ , the function  $G[\pi]$ , as a function on  $S_{2(n-1), n-1}$ , is pseudo-random in the sense of Definition 21 (we stress again that  $\pi(2n - 1) = \pi(2n) = n$  is pre-defined). Moreover, the degree of  $G[\pi]$  is (at most)  $2d$ . The subtle explanation is as follows.<sup>4</sup> By definition,  $G[\pi]$  is the average of

$$(g(\pi^{-1}(x)) - g(\pi^{-1}(x + s e)))^2 = g(\pi^{-1}(x))^2 + g(\pi^{-1}(x + s e))^2 - 2g(\pi^{-1}(x))g(\pi^{-1}(x + s e)) \quad (3)$$

over some distribution over  $x, s$ , so it is enough to argue about the degree for each fixed  $x, s$ . If either of the inputs  $\pi^{-1}(x)$  or  $\pi^{-1}(x + s e)$  falls outside of the set  $E$ , their  $g$ -value is zero and can be dropped from consideration. Otherwise their  $g$ -values are given by the degree- $d$  function  $f^{\leq d} : [m]^{2n} \rightarrow \mathbb{R}$ . Thus (3) can be written as a linear combination of monomials of degree at most  $2d$  and any monomial, say on coordinates  $i_1, \dots, i_{2d}$ , is determined by  $\pi(i_1), \dots, \pi(i_{2d})$  when regarded as a function on  $S_{2(n-1), n-1}$ .

Thus  $G[\pi]$  is a degree- $2d$  pseudo-random function and we can apply Lemma 22 to upper bound its fourth moment as  $\mathbb{E}_{\pi} [G[\pi]^4] \leq C(d, m) \mathbb{E}_{\pi} [G[\pi]^2]^2$ . Finally, by  $(\frac{3}{2}, 3)$ -Holder's inequality,  $\mathbb{E}_{\pi} [G[\pi]^2] = \mathbb{E}_{\pi} [G[\pi]^{\frac{2}{3}} \cdot G[\pi]^{\frac{4}{3}}] \leq (\mathbb{E}_{\pi} [G[\pi]])^{\frac{2}{3}} (\mathbb{E}_{\pi} [G[\pi]^4])^{\frac{1}{3}}$ , which yields, using the bound on the fourth moment,  $\mathbb{E}_{\pi} [G[\pi]^2] \leq C(d, m) \mathbb{E}_{\pi} [G[\pi]]^2$ , and then  $\mathbb{E}_{\pi} [G[\pi]^4] \leq C(d, m) \mathbb{E}_{\pi} [G[\pi]]^4$ . Using Markov and Lemma 31, we conclude as desired, that

$$\Pr_{\pi} [G[\pi] \geq \delta] \leq \frac{\mathbb{E}_{\pi} [G[\pi]^4]}{\delta^4} \leq \frac{C(d, m)}{\delta^4} \mathbb{E}_{\pi} [G[\pi]]^4 \leq \frac{C(d, K, m)}{\delta^4} (I_{2n-1}[g]^4 + I_{2n}[g]^4). \blacktriangleleft$$

► **Lemma 35.** *Let  $f : [m]^{2n} \rightarrow [0, 1]$  be a bounded function and  $E \subseteq [m]^{2n}$  be the set of  $K$ -roughly balanced inputs. Define  $g = f^{\leq d} \cdot 1_E$  as in the statement of Lemma 34. Then*

$$\Pr_{\pi} [\exists j : I_j[g|\pi] \geq \delta] \leq \frac{C(d, K, m)}{\delta^4} \cdot \sum_{i=1}^{2n} I_i[g]^4.$$

**Proof.** We use Lemma 34, but note that the consideration of the  $n^{th}$  coordinate and the requirement that  $\pi(2n - 1) = \pi(2n) = n$  is only for notational convenience. What we have actually proved is that for any  $1 \leq j \leq n$  and any  $1 \leq i \neq i' \leq 2n$ ,

$$\Pr_{\pi: \pi(i) = \pi(i') = j} [I_j[g|\pi] \geq \delta] \leq \frac{C(d, K, m)}{\delta^4} (I_i[g]^4 + I_{i'}[g]^4).$$

<sup>4</sup> To be in strict accordance with Definition 21, one actually argues here that  $G[\pi]$  has degree  $d^* = 2d$  and the pseudo-randomness condition holds for all  $|A| = 4d^*, |B| = d^*$ . We have avoided this minor point for ease of presentation.

Fixing  $j$  and taking average over all  $1 \leq i \neq i' \leq 2n$  gives

$$\Pr_{\pi} [I_j[g|\pi] \geq \delta] \leq \frac{C(d, K, m)}{\delta^4} \mathbb{E}_{\pi} \left[ \sum_{i \in \pi^{-1}(j)} I_i[g]^4 \right].$$

Now taking a union bound over all  $1 \leq j \leq n$  gives the result.  $\blacktriangleleft$

The above Lemma 35 shows, morally speaking, that with high probability over the choice of  $\pi$ , all influences of  $g|_{\pi}$  are low provided that all influences of  $g$  are low. We could upper bound  $\sum_{i=1}^{2n} I_i[g]^4$  by  $\tau(g)^3 I[g]$  where  $\tau(g)$  is the maximum influence  $I_i[g]$  and  $I[g] = \sum_{i=1}^{2n} I_i[g]$  is the total influence. The total influence, since  $g$  is morally speaking same as  $f^{\leq d}$ , should be  $O(d)$ . However, the fact that  $g = f^{\leq d} 1_E$  is a truncation of  $f^{\leq d}$  complicates matters and we have to go through a somewhat tedious argument.

#### 4.4 Relating Influences of $f$ and $g$

► **Lemma 36.** *Let  $f : [m]^{2n} \rightarrow [0, 1]$  be a bounded function and  $E \subseteq [m]^{2n}$  be the set of  $K$ -roughly balanced inputs. Define  $g = f^{\leq d} \cdot 1_E$ . Then for any coordinate  $1 \leq i \leq 2n$ ,  $I_i[g] \leq C(m) I_i^{\leq d}[f] + C(d, m) n^{-\frac{3}{8}}$ .*

**Proof.** By definition,  $g = f^{\leq d} \cdot 1_E$  and  $I_i[g]$  equals (possibly up to a factor  $m$ )

$$\mathbb{E}_{z \in_R [m]^{2n}} [|f^{\leq d}(z) \cdot 1_E(z) - f^{\leq d}(z + e_i) \cdot 1_E(z + e_i)|^2].$$

Now if both  $z$  and  $z + e_i$  are in  $E$ , the term inside is same as  $|f^{\leq d}(z) - f^{\leq d}(z + e_i)|^2$  and it contributes to the influence  $I_i^{\leq d}[f]$ . So only additional contribution to  $I_i[g]$  on top of  $I_i^{\leq d}[f]$  is due to inputs  $z$  such that  $z \in E$ , but  $z + e_i \notin E$  (or vice versa). Let  $\partial E$  denote the set of such  $z$  so that it constitutes at most  $\frac{C(m)}{\sqrt{n}}$  fraction of inputs in  $[m]^{2n}$ . The additional contribution to  $I_i[g]$  is now upper bounded as (using  $(4, \frac{4}{3})$ -Holder)

$$\mathbb{E} [f^{\leq d}(z)^2 1_{\partial E}] \leq \mathbb{E} [f^{\leq d}(z)^8]^{\frac{1}{4}} \mathbb{E} [1_{\partial E}^{\frac{4}{3}}]^{\frac{3}{4}} \leq C(d, m) \left( \frac{C(m)}{\sqrt{n}} \right)^{\frac{3}{4}}.$$

We used  $\mathbb{E} [f^{\leq d}(z)^8] \leq C(d, m) \mathbb{E} [f^{\leq d}(z)^2]^4 \leq C(d, m)$  that follows from Theorem 11. This completes the proof.  $\blacktriangleleft$

► **Lemma 37.** *Let  $f : [m]^{2n} \rightarrow [0, 1]$  be a bounded function and  $E \subseteq [m]^{2n}$  be the set of  $K$ -roughly balanced inputs. Define  $g = f^{\leq d} \cdot 1_E$ . Then except with probability  $\zeta$  over the choice of  $\pi$ , we have*

$$\max_{1 \leq j \leq n} I_j[f|\pi] \leq 3 \cdot \max_{1 \leq j \leq n} I_j[g|\pi] + \delta.$$

This holds as long as  $K = O(\log \frac{1}{\delta \zeta})$  is sufficiently large and  $\|f^{>d}\|_2^2 \leq \gamma = \gamma(m, \delta, \zeta)$  is sufficiently small. We emphasize that  $\gamma$  does not depend on  $d$ .

**Proof.** We write  $f = g + h + q$  where  $g = f^{\leq d} \cdot 1_E$ ,  $h = f^{>d} \cdot 1_E$ , and  $q = f \cdot 1_{\bar{E}}$ . Clearly, for any coordinate  $1 \leq j \leq n$  (using  $(a + b + c)^2 \leq 3(a^2 + b^2 + c^2)$ ),

$$I_j[f|\pi] \leq 3 \cdot (I_j[g|\pi] + I_j[h|\pi] + I_j[q|\pi]).$$

We will show that except with “small” probability over the choice of  $\pi$ , both  $\|h|_{\pi}\|_2^2$  and  $\|q|_{\pi}\|_2^2$  are “small”. Since these are upper bounds on  $I_j[h|\pi]$  and  $I_j[q|\pi]$  respectively, the lemma follows. We will just show that  $\mathbb{E}_{\pi} [\|h|_{\pi}\|_2^2]$  and  $\mathbb{E}_{\pi} [\|q|_{\pi}\|_2^2]$  are “small” (i.e.  $\ll \delta \zeta$ ) and this determines the quantitative constraints on  $K$  and  $\gamma$  and then use Markov. Indeed,

## 27:16 On Rich 2-to-1 Games

- Towards upper-bounding  $\mathbb{E}_\pi [\|q|_\pi\|_2^2]$ , we note that  $f$  is bounded in  $[0, 1]$  and

$$\mathbb{E}_\pi [\|q|_\pi\|_2^2] = \mathbb{E}_{\pi, x \in [m]^n} [f(\pi^{-1}(x)) \cdot 1_{\overline{E}}(\pi^{-1}(x))] \leq \mathbb{E}_{\pi, x \in [m]^n} [1_{\overline{E}}(\pi^{-1}(x))],$$

and the probability that  $\pi^{-1}(x)$  is imbalanced is at most  $2^{-\Omega(K)}$ .

- Towards upper-bounding  $\mathbb{E}_\pi [\|h|_\pi\|_2^2]$ , we argue that ( $z = \pi^{-1}(x)$ )

$$\mathbb{E}_\pi [\|h|_\pi\|_2^2] = \mathbb{E}_{\pi, x \in [m]^n} [f^{>d}(z)^2 1_E(z)] \leq C(K, m) \|f^{>d}\|_2^2 \leq C(K, m) \gamma.$$

In the second step, since one is concerned only with  $K$ -roughly balanced inputs, one can “switch” to uniform distribution over input thanks to Lemma 27. ◀

### Proof of Lemma 24

We now complete the proof of Lemma 24. Let  $f : [m]^{2n} \rightarrow [0, 1]$  be a function as therein with  $\|f^{>d}\|_2^2 \leq \gamma$  and for all  $i \in [2n]$ ,  $I_i^{\leq d}[f] \leq \tau$ . Let  $g = f^{\leq d} 1_E$  where  $E$  is the set of  $K$ -roughly balanced inputs. The parameters  $K, \gamma, \tau$  are chosen as needed by the proof.

By Lemma 36, we get an upper bound as below. We note that the total influence of  $f^{\leq d}$  is  $O(d)$  and its maximum influence is at most  $\tau$  by hypothesis.

$$\sum_{i=1}^{2n} I_i[g]^4 \leq C(d, m) \left( \sum_{i=1}^{2n} I_i^{\leq d}[f]^4 + \frac{1}{\sqrt{n}} \right) \leq C(d, m) \tau^3.$$

This gives, by Lemma 35, that  $\Pr_\pi [\exists j : I_j[g|_\pi] \geq \delta] \leq \frac{C(d, K, m)}{\delta^4} \tau^3$ . Finally, by Lemma 37, except with probability  $\zeta$  over the choice of  $\pi$ , it holds that  $\max_{1 \leq j \leq n} I_j[f|_\pi] \leq 3 \cdot \max_{1 \leq j \leq n} I_j[g|_\pi] + \delta$ . Putting the two conclusions together, we conclude that

$$\Pr_\pi [\exists j : I_j[f|_\pi] \geq 4\delta] \leq \zeta + \frac{C(d, K, m)}{\delta^4} \tau^3 \leq 2\zeta,$$

completing the proof of Lemma 24. In terms of quantitative constraints on the parameters,  $K = K(\delta, \zeta)$ ,  $\gamma = \gamma(m, \delta, \zeta)$  are determined by Lemma 35 and  $\tau = \tau(d, m, \delta, \zeta)$  needs to obey the very last inequality above.

## 5 The Reduction

We now prove Theorem 8 that the Rich 2-to-1 Games Conjecture is equivalent to the Unique Games Conjecture. The reduction from Unique Games to Rich 2-to-1 Games as well as its analysis are standard and appear in the full version. The reduction from Rich 2-to-1 Games to Unique Games is also standard and is presented in this section. Its analysis however needs new analytic tools, specifically Lemma 23.

We are given a Rich 2-to-1 Games instance  $\Psi = (L \cup R, E, \Sigma_L, \Sigma_R, \Phi)$  with  $\Sigma_L = [2n]$ ,  $\Sigma_R = [n]$ , completeness (at least)  $1 - \eta$ , and soundness (at most)  $\eta$ . The reduction outputs an instance of Unique Games with alphabet  $[m]$ , completeness (at least)  $1 - 5\varepsilon$ , and soundness (at most)  $\varepsilon$ . For given  $\varepsilon$ , first  $m$  needs to be taken sufficiently large, then  $\eta$  sufficiently small, and in turn  $n$  sufficiently large. The instance of Unique Games produced is linear, i.e. its alphabet  $[m]$  is identified with  $\mathbb{Z}_m$ , the additive group of integers modulo  $m$ , and the constraints are linear equations.

As is standard, we replace a vertex  $u \in L$  with the (supposed)  $m$ -ary long code of the (supposed) label of  $u$ . The positions in the long code correspond to the variables of the Unique Games instance. An assignment to these variables corresponds to a function  $F_u : [m]^{2n} \rightarrow [m]$ . The intention is that if  $i \in [2n]$  is the label of  $u$  (in the 2-to-1 Games instance), then  $F_u(x) = F_u(x_1, \dots, x_{2n}) = x_i$  is the corresponding dictatorship function. In our reduction, the long codes for labels of vertices  $v \in R$  do not appear explicitly, but it will be convenient to imagine them “virtually”.

The PCP test is straightforward: one performs a two-query “noise-test” on the virtual long code of a vertex  $v \in R$ , but actually reads off the queries from the long codes of neighbors  $u, w$  of the vertex  $v$  respectively (the virtual long code for  $v$  is “contained” in that of  $u$  as well as  $w$ ). Each test is viewed as a Unique Games constraint and this defines the Unique Games instance produced by the reduction. Formally, a test/equation is produced as follows:

- Sample  $v \in R$  uniformly,  $a \in [m]^n$  at random, and  $b \in [m]^n$  to be  $1 - \varepsilon$  correlated with  $a$ .
- Sample two neighbours  $u, w$  of  $v$  independently at random.
- Set  $A = \pi_{(u,v)}^{-1}(a)$ ,  $B = \pi_{(w,v)}^{-1}(b) \in [m]^{2n}$ .
- Finally, sample  $x \in [m]^{2n}$  that is  $1 - \varepsilon$  correlated with  $A$ , and  $y \in [m]^{2n}$  that is  $1 - \varepsilon$  correlated with  $B$ . Output the equation  $F_u(x) = F_w(y)$ .

## Folding

As is standard, we can assume that the functions  $F_u : [m]^{2n} \rightarrow [m]$  that appear in the PCP proof are folded, meaning  $F_u(x + se) = F_u(x) + s$  where  $e \in [m]^{2n}$  is the all 1 vector. In particular,  $F_u$  is then balanced, i.e. takes all values in  $[m]$  equally often. Technically, folding is enforced by keeping only one of the inputs in the set  $\{x + se \mid s \in [m]\}$  as a representative and inferring values at other inputs from the representative. The effect of folding is that the equations produced are of the type  $p = q + s$  instead of just  $p = q$  where  $p, q$  are the Unique Games variables in the output instance and  $s \in [m]$ .

## 5.1 Completeness

If the 2-to-1 Games instance  $\Psi$  has a labeling  $\sigma : L \rightarrow [2n]$ ,  $\rho : R \rightarrow [n]$  that satisfies at least  $1 - \eta$  fraction of the constraints, we show that the Unique Games instance is (at least)  $1 - 2\eta - 3\varepsilon \geq 1 - 5\varepsilon$  satisfiable for  $\eta$  sufficiently small.

Indeed, define for any  $u \in L$ , the long-code assignment  $F_u(x) = x_{\sigma(u)}$ . Since the edges  $(u, v), (w, v)$  are distributed uniformly, with probability at least  $1 - 2\eta$ , both edges are satisfied by the labeling, i.e.  $\pi_{(u,v)}(\sigma(u)) = \rho(v) = \pi_{(w,v)}(\sigma(w))$ . Whenever this happens, the test accepts with probability at least  $1 - 3\varepsilon$  since the failure to accept can be attributed to one of three events: strings  $a$  and  $b$  differing on the co-ordinate  $\rho(v)$ , strings  $x$  and  $A$  differing on the co-ordinate  $\sigma(u)$ , or strings  $y$  and  $B$  differing on the co-ordinate  $\sigma(w)$ .

## 5.2 Soundness

We will show that if the 2-to-1 Games instance has soundness at most  $\eta$  (to be chosen sufficiently small later), then the probability that the test accepts is upper bounded by  $\varepsilon$ .

Let  $F_u : [m]^{2n} \rightarrow [m]$  be the folded functions given as assignment to the Unique Games instance. In a standard manner, we view the functions as  $F_u : [m]^{2n} \rightarrow \Delta_m$  where  $\Delta_m = \{(t_0, \dots, t_{m-1}) \mid t_i \geq 0, \sum_{i=0}^{m-1} t_i = 1\}$  is the standard  $m$ -dimensional simplex. Each function  $F_u$  is then thought of as a vector  $(F_{u,0}, \dots, F_{u,m-1})$  where each  $F_{u,r}$  is a  $\{0, 1\}$ -valued function and  $\mathbb{E}[F_{u,r}] = \frac{1}{m}$  since  $F_u$  is folded and balanced. Moreover, the acceptance criterion

of the test, i.e.  $F_u(x) = F_w(y)$ , can be written arithmetically as  $\sum_{r=0}^{m-1} F_{u,r}(x) \cdot F_{w,r}(y)$ . Hence the probability that the test accepts can be written as (the expectation is over all the choices made)

$$\mathbb{E}_{v,u,w,a,b,A,B,x,y} \left[ \sum_{r=0}^{m-1} F_{u,r}(x) \cdot F_{w,r}(y) \right] = \sum_{r=0}^{m-1} \mathbb{E}_{v,u,w,a,b,A,B,x,y} [F_{u,r}(x) \cdot F_{w,r}(y)]. \quad (4)$$

Henceforth we will fix the index  $0 \leq r \leq m-1$  and then show an upper bound on the expectation on the right (with the overall upper bound being  $m$  times that). For notational convenience, we drop the subscript  $r$  and define  $\{0,1\}$ -valued functions  $f_u = F_{u,r}$ . Thus the goal is to upper bound (note that  $a, b \in [m]^n$  and  $A, B, x, y \in [m]^{2n}$ )

$$\begin{aligned} \mathbb{E}_{\substack{v,u,w \\ a \sim_{1-\varepsilon} b}} \left[ \mathbb{E}_{x \sim_{1-\varepsilon} A, y \sim_{1-\varepsilon} B} [f_u(x) \cdot f_w(y)] \right] &= \mathbb{E}_{\substack{v,u,w \\ a \sim_{1-\varepsilon} b}} [T_{1-\varepsilon} f_u(A) \cdot T_{1-\varepsilon} f_w(B)] \\ &= \mathbb{E}_{\substack{v,u,w \\ a \sim_{1-\varepsilon} b}} [g_u(A) \cdot g_w(B)], \end{aligned}$$

where  $g_u = T_{1-\varepsilon} f_u$ . We note that  $g_u$  is  $[0,1]$ -valued and  $\mathbb{E}[g_u] = \mathbb{E}[f_u] = \frac{1}{m}$ . We further define  $g_{u,v} = g_u|_{\pi(u,v)}$  and we still have  $\mathbb{E}[g_{u,v}] = \frac{1}{m}$ . The expectation can be rewritten as

$$\mathbb{E}_{\substack{v,u,w \\ a \sim_{1-\varepsilon} b}} [g_{u,v}(a) \cdot g_{w,v}(b)] = \mathbb{E}_{v,u,w} [\langle g_{u,v}, T_{1-\varepsilon} g_{w,v} \rangle] = \mathbb{E}_v [\langle h_v, T_{1-\varepsilon} h_v \rangle], \quad (5)$$

where in the last step we used the fact that the choices of  $u, w$  are independent (for a fixed  $v$ ) and defined  $h_v = \mathbb{E}_u [g_{u,v}]$ . We note that  $\mathbb{E}[h_v] = \frac{1}{m}$  as well. We now show, by way of contradiction, that if the expectation in (5) is at least  $\beta = \frac{\varepsilon}{m}$ , then one can define a labeling to the 2-to-1 Games instance that satisfies more than  $\eta$  fraction of its constraints. It then follows that (5) is bounded by  $\beta$  and hence (4) (i.e. the acceptance probability of PCP test) by  $m\beta = \varepsilon$  as desired.

Assume therefore that the expectation in (5) is at least  $\beta$ . By an averaging argument, for at least  $\frac{\beta}{2}$  fraction of vertices  $v \in R$ , the inner product  $\langle h_v, T_{1-\varepsilon} h_v \rangle$  is at least  $\frac{\beta}{2}$ . Let  $R_{\text{Good}} \subseteq R$  be the subset of such vertices. That is, for  $v \in R_{\text{Good}}$ ,  $\langle h_v, T_{1-\varepsilon} h_v \rangle \geq \frac{\beta}{2}$ . Using Theorem 16, the function  $h_v$  then must have an influential co-ordinate, and moreover since  $h_v = \mathbb{E}_u [g_{u,v}]$ , so does the function  $g_{u,v}$  for a good fraction of the neighbors  $u \in L$ . In light of this observation, we hope to come up with a labeling to vertices  $v \in R$  and  $u \in L$  by choosing an influential co-ordinate of the function  $h_v$  and the function  $g_u$  respectively (we need to use the fact that  $g_u$  is smooth or low-degree). This strategy works thanks to our main technical Lemma 23.

Indeed, for  $v \in R_{\text{Good}}$ , define its label  $\rho(v)$  to be an arbitrary co-ordinate  $j \in [n]$  such that  $I_j[h_v] \geq \delta$ . Such a coordinate exists since  $\langle h_v, T_{1-\varepsilon} h_v \rangle \geq \frac{\beta}{2}$  and using Theorem 16. One needs to take  $m$  sufficiently large so that  $\mathbb{E}[h_v] = \frac{1}{m} = \theta$  is sufficiently small to bring the bound  $2\Gamma_{1-\varepsilon}(\theta)$  in Theorem 16 below  $\frac{\beta}{2} = \frac{\varepsilon}{2} \frac{1}{m} = \frac{\varepsilon}{2} \theta$ . One then needs to take the influence parameter  $\delta$  therein sufficiently small.

Since  $h_v = \mathbb{E}_u [g_{u,v}]$  and  $\rho(v)$  has influence at least  $\delta$  on  $h_v$ , it follows that for at least  $\frac{\delta}{2}$  fraction of neighbors  $u \in L$  of  $v$  we have  $I_{\rho(v)}[g_{u,v}] \geq \frac{\delta}{2}$ . Let  $N_{\text{Good}}(v)$  denote the subset of such neighbors. We emphasize that for a random choice of edge  $(u, v)$ , we have  $v \in R_{\text{Good}}$  and  $u \in N_{\text{Good}}(v)$  with probability at least  $\frac{\beta}{2} \frac{\delta}{2}$ .



Now, by the main Lemma 23, except with probability  $\zeta \ll \frac{\beta\delta}{8}$  over the choice of the edge  $(u, v)$ , it is the case that whenever  $I_j[g_{u,v}] \geq \frac{\delta}{2}$  for some  $j \in [n]$ , one has  $I_i[g_u] \geq \tau$  for some  $i \in \pi^{-1}(j), \pi = \pi_{(u,v)}$ . We note that since  $g_u = T_{1-\varepsilon}f_u$ , its Fourier mass beyond degree  $d$  is at most  $\gamma = 2^{-\Omega(d/\varepsilon)}$ , which can be made sufficiently small by taking  $d$  sufficiently large. Finally,  $\tau$  is taken to be sufficiently small so that the lemma applies. It follows that with probability  $\frac{\beta\delta}{8}$ , all these events happen simultaneously:

$$v \in R_{\text{Good}}, \quad u \in N_{\text{Good}}(v), \quad I_{\rho(v)}[g_{u,v}] \geq \frac{\delta}{2}, \quad I_i^{\leq d}[g_u] \geq \tau \quad \text{for some } i \in \pi^{-1}(\rho(v)).$$

Thus if we defined a label for  $u \in L$  by making a list of all co-ordinates with degree- $d$  influence at least  $\tau$  on  $g_u$  and then picked one label at random from this list, it would agree with  $\rho(v)$  (via  $\pi = \pi_{(u,v)}$ ) with probability at least  $\Omega(\frac{\tau}{d})$  (the list size is  $O(\frac{d}{\tau})$  since the total degree- $d$  influence is at most  $d$ ). This gives a labeling to the 2-to-1 Games instance that satisfies overall  $\Omega(\frac{\beta\delta\tau}{d})$  fraction of its constraints. Choosing the soundness  $\eta$  of the 2-to-1 Games instance to be even lower a priori completes the proof.

---

## References

- 1 Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *J. ACM*, 45(3):501–555, 1998.
- 2 Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: A new characterization of NP. *J. ACM*, 45(1):70–122, 1998.
- 3 Boaz Barak, Pravesh K. Kothari, and David Steurer. Small-set expansion in shortcode graph and the 2-to-2 conjecture. In *10th Innovations in Theoretical Computer Science Conference, ITCS 2019, January 10-12, 2019, San Diego, California, USA*, pages 9:1–9:12, 2019.
- 4 Amey Bhangale, Subhash Khot, and Devanathan Thiruvengatachari. An improved dictatorship test with perfect completeness. In *37th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2017, December 11-15, 2017, Kanpur, India*, pages 15:1–15:23, 2017.
- 5 Irit Dinur, Subhash Khot, Guy Kindler, Dor Minzer, and Muli Safra. On non-optimally expanding sets in grassmann graphs. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 940–951, 2018.
- 6 Irit Dinur, Subhash Khot, Guy Kindler, Dor Minzer, and Muli Safra. Towards a proof of the 2-to-1 games conjecture? In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 376–389, 2018.
- 7 Irit Dinur, Elchanan Mossel, and Oded Regev. Conditional hardness for approximate coloring. *SIAM J. Comput.*, 39(3):843–873, 2009.
- 8 Uriel Feige, Shafi Goldwasser, Laszlo Lovász, Shmuel Safra, and Mario Szegedy. Interactive proofs and the hardness of approximating cliques. *J. ACM*, 43(2):268–292, March 1996.
- 9 Uriel Feige and László Lovász. Two-prover one-round proof systems: Their power and their problems (extended abstract). In *Proceedings of the 24th Annual ACM Symposium on Theory of Computing, May 4-6, 1992, Victoria, British Columbia, Canada*, pages 733–744, 1992.
- 10 Yuval Filmus, Guy Kindler, Noam Lifshitz, and Dor Minzer. Hypercontractivity on the symmetric group. *arXiv preprint*, 2020. [arXiv:2009.05503](https://arxiv.org/abs/2009.05503).
- 11 Subhash Khot. On the power of unique 2-prover 1-round games. In *Proceedings of the 17th Annual IEEE Conference on Computational Complexity, Montréal, Québec, Canada, May 21-24, 2002*, page 25, 2002.
- 12 Subhash Khot. Inapproximability of NP-complete problems, discrete fourier analysis, and geometry. In *Proceedings of the International Congress of Mathematicians 2010*, pages 2676–2697, 2010.

- 13 Subhash Khot. On the unique games conjecture. In *Proceedings of the 25th Annual IEEE Conference on Computational Complexity, CCC 2010, Cambridge, Massachusetts, June 9-12, 2010*, pages 99–121, 2010.
- 14 Subhash Khot, Guy Kindler, Elchanan Mossel, and Ryan O’Donnell. Optimal inapproximability results for max-cut and other 2-variable csps? *SIAM J. Comput.*, 37(1):319–357, April 2007.
- 15 Subhash Khot, Dor Minzer, Dana Moshkovitz, and Muli Safra. Small set expansion in the johnson graph. *Electronic Colloquium on Computational Complexity (ECCC)*, 25:78, 2018.
- 16 Subhash Khot, Dor Minzer, and Muli Safra. On independent sets, 2-to-2 games, and grassmann graphs. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 576–589, 2017.
- 17 Subhash Khot, Dor Minzer, and Muli Safra. Pseudorandom sets in grassmann graph have near-perfect expansion. In *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 592–601, 2018.
- 18 Subhash Khot and Oded Regev. Vertex cover might be hard to approximate to within 2-epsilon. *J. Comput. Syst. Sci.*, 74(3):335–349, 2008.
- 19 Subhash Khot and Muli Safra. A two-prover one-round game with strong soundness. *Theory of Computing*, 9(28):863–887, 2013. doi:10.4086/toc.2013.v009a028.
- 20 Ran Raz. A parallel repetition theorem. *SIAM J. Comput.*, 27(3):763–803, June 1998.
- 21 Suguru Tamaki and Yuichi Yoshida. A query efficient non-adaptive long code test with perfect completeness. *Random Struct. Algorithms*, 47(2):386–406, 2015.
- 22 Luca Trevisan. On Khot’s unique games conjecture. *Bulletin of the AMS*, 49(1):91–111, 2012.

## A Standard Facts and Calculations

In this section we collect several standard statements; the proofs can be found in the full version of the paper.

► **Lemma 38.** *For any positive integers  $r, t$ , there are constants  $0 < c(r, t) < C(r, t)$  such that the following holds for large enough  $n$ . Let  $v_1, \dots, v_r \geq \frac{n}{10r}$  be integers that sum up to  $n$ , let  $u_1, \dots, u_r$  be non-negative integers that are each at most  $t$ , and denote  $u = u_1 + \dots + u_r$ . Then  $c(r, t) \binom{n}{v_1, \dots, v_r} \leq \binom{n-u}{v_1-u_1, \dots, v_r-u_r} \leq C(r, t) \binom{n}{v_1, \dots, v_r}$ .*

► **Lemma 39.** *For any positive integer  $r$ , there is a constant  $C(r) > 0$  such that the following holds for large enough  $n$ . Let  $v_1, \dots, v_r$  be integers of absolute value at most  $\sqrt{K \cdot r \cdot n}$  that sum to zero, and such that for every  $1 \leq i \leq r$ , the integer  $n + v_i$  is divisible by  $r$ . Then*

$$\frac{\binom{(n+v_1)/r, \dots, (n+v_r)/r}{2n}}{\binom{2(n+v_1)/r, \dots, 2(n+v_r)/r}} \leq C(r) 2^{Kr} r^{-n}.$$

Let  $A_1, \dots, A_r$  be a partition of  $[2n]$  into  $r$  even-sized sets. We say a mapping  $\pi \in S_{2n,n}$  is consistent with  $A_1, \dots, A_r$  if matching given by  $\pi$  matches off each set  $A_i$  within itself (or equivalently that  $\pi^{-1}(\pi(A_i)) = A_i$ ).

► **Lemma 40.** *Let  $A_1, \dots, A_r$  be a partition of  $[2n]$  into even-sized sets, and denote their sizes by  $a_1, \dots, a_r$ . Then  $\Pr_{\pi \in S_{2n,n}} [\pi \text{ is consistent with } A_1, \dots, A_r] = \frac{\binom{a_1}{2}, \dots, \binom{a_r}{2}}{\binom{2n}{2}}.$*

# Distributed Quantum Proofs for Replicated Data

**Pierre Fraigniaud**

IRIF, CNRS and Université de Paris, France

**François Le Gall**

Graduate School of Mathematics, Nagoya University, Japan

**Harumichi Nishimura**

Graduate School of Informatics, Nagoya University, Japan

**Ami Paz**

Faculty of Computer Science, Universität Wien, Austria

---

## Abstract

This paper tackles the issue of *checking* that all copies of a large data set replicated at several nodes of a network are identical. The fact that the replicas may be located at distant nodes prevents the system from verifying their equality locally, i.e., by having each node consult only nodes in its vicinity. On the other hand, it remains possible to assign *certificates* to the nodes, so that verifying the consistency of the replicas can be achieved locally. However, we show that, as the replicated data is large, classical certification mechanisms, including distributed Merlin-Arthur protocols, cannot guarantee good completeness and soundness simultaneously, unless they use very large certificates. The main result of this paper is a distributed *quantum* Merlin-Arthur protocol enabling the nodes to collectively check the consistency of the replicas, based on small certificates, and in a single round of message exchange between neighbors, with short messages. In particular, the certificate-size is logarithmic in the size of the data set, which gives an exponential advantage over classical certification mechanisms. We propose yet another usage of a fundamental quantum primitive, called the SWAP test, in order to show our main result.

**2012 ACM Subject Classification** Theory of computation → Quantum communication complexity; Theory of computation → Distributed computing models

**Keywords and phrases** Quantum Computing, Distributed Network Computing, Algorithmic Aspects of Networks

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.28

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/2002.10018>.

**Funding** *Pierre Fraigniaud*: Additional support from the ANR projects *DESCARTES* and *QuDATA*. *François Le Gall*: JSPS KAKENHI grants Nos. JP16H01705, JP19H04066, JP20H04139 and JP20H00579 and MEXT Quantum Leap Flagship Program Grant Number JPMXS0120319794.

*Harumichi Nishimura*: JSPS KAKENHI grants Nos. JP16H01705, JP19H04066 and MEXT Quantum Leap Flagship Program Grant Number JPMXS0120319794.

*Ami Paz*: We acknowledge the Austrian Science Fund (FWF) and netIDEE SCIENCE project P 33775-N.

**Acknowledgements** We thank the anonymous reviewer of ITCS 2021 who pointed [44] to us, and Uri Meir for useful discussions.

## 1 Introduction

In the context of distributed systems, the presence of faults potentially corrupting the individual states of the nodes creates a need to regularly check whether the system is in a global state that is legal with respect to its specification. A basic example is a system storing data, and using replicas in order to support crash failures. In this case, the application managing the data is in charge of regularly checking that the several replicas of the same



© Pierre Fraigniaud, François Le Gall, Harumichi Nishimura, and Ami Paz;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 28; pp. 28:1–28:20



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

data, stored at different nodes scattered in the network, are all identical. Another example is an application maintaining a tree spanning the nodes of a network, e.g., for multicast communication. In this case, every node stores a pointer to its parent in the tree, and the application must regularly check that the collection of pointers forms a spanning tree. This paper addresses the issue of checking the correctness of a distributed system configuration at low cost.

Several mechanisms have been designed for certifying the correctness of the global state of a system in a distributed manner. One popular mechanism is called *locally checkable proofs* [24], and it extends the seminal concept of *proof-labeling schemes* [35]. In these frameworks, the distributed application does not only construct or maintain some distributed data structure (e.g., a spanning tree), but also constructs a distributed *proof* that the data structure is correct. This proof has the form of a *certificate* assigned to each node (the certificates assigned to different nodes do not need to be the same). For collectively checking the legality of the current global system state, the nodes exchange their certificates with their neighbors in the network. Then, based on its own individual state, its certificate, and the certificates of its neighbors, every node accepts or rejects, according to the following specification. If the global state is legal, and if the certificates are assigned properly by the application, then all nodes accept. Conversely, if the global state is illegal, then at least one node rejects, *no matter which certificates are assigned to the nodes*. Such a rejecting node can raise an alarm, or launch a recovery procedure. The main aim of locally checkable proofs is to be *compact*, that is, to use certificates as small as possible, for two reasons: first, to limit the space complexity at each node, and, second, to limit the message complexity of the verification procedure involving communications between neighbors.

For instance, in the case of the Spanning Tree predicate, the application does not only construct a spanning tree  $T$  of the network, but also a distributed proof that  $T$  is indeed a spanning tree, i.e., that the collection  $T$  of pointers forms a cycle-free connected spanning subgraph. It has been known for long [2, 6, 26] that, by assigning to every node a certificate of logarithmic size, the nodes can collectively check whether  $T$  is indeed a spanning tree, in a single round of communication between neighboring nodes. The certificate assigned to a node is the identity of the root of the tree, and its distance to this root (both are of logarithmic size as long as the IDs are in a range polynomial in the number of nodes). Every node just checks that it is provided with the same root-ID as all its neighbors in the network, and that the distance given to its parent in its certificate is one less than its own given distance – a node with distance 0 checks that its ID is indeed the root-ID provided in its certificate. Obviously, if the collection  $T$  of pointers forms a spanning tree, and if the certificates are assigned properly by the application, then all nodes pass these tests, and accept. On the other hand, it is easy to check that if  $T$  is not a spanning tree (it is not connected, or it contains a cycle), then at least one node detects a discrepancy and rejects, no matter which certificates are assigned to the nodes.

Unfortunately, not all boolean predicates on labeled graphs can be distributedly certified using certificates as small as for spanning tree. This is typically the case of the aforementioned scenario of a distributed data storage using replicas, for which one must certify equality. Let us for instance consider the case of two nodes Alice and Bob at the two extremities of a path, that is, the two players are separated by intermediate nodes. Alice and Bob respectively store two  $n$ -bit strings  $x$  and  $y$ , and the objective is to certify that  $x = y$ . That is, one wants to certify equality (EQ) between *distant* players. A direct reduction from the non-deterministic communication complexity of EQ shows that certifying EQ cannot be achieved with certificates smaller than  $\Omega(n)$  bits.

Randomization may help circumventing the difficulty of certifying some boolean predicates on labeled graphs using small certificates. Hence, a weaker form of protocols has been considered, namely *distributed Merlin-Arthur* protocols (dMA), a.k.a. *randomized proof-labeling schemes* [22]. In this latter context, Merlin provides the nodes with a proof, just like in locally checkable proofs, and Arthur performs a *randomized* local verification at each node. Unfortunately, some predicates remain hard in this framework too. In particular, as we show in the paper, there are no classical dMA protocols for (distant) EQ using compact certificates. Recently, several extensions of dMA protocols were proposed, e.g., by allowing more interaction between the prover and the verifier [15, 21, 39]. In this work, we add the quantum aspect, while considering only a single interaction, and only in the prescribed order: Merlin sends a proof to Arthur, and then there is no more interaction between them.

## 1.1 Our Results

We carry on the recent trend of research consisting of investigating the power of quantum resources in the context of distributed network computing (cf., e.g., [17, 37, 27, 38, 28, 23]), by designing a distributed Quantum Merlin-Arthur (dQMA) protocol for distant EQ, using compact certificates and small messages. While we use the dQMA terminology in order to be consistent with prior work, we emphasize that the structure of the discussed protocols is rather simple: each node is given a quantum state as a certificate, the nodes exchange these states, perform a local computation, and finally accept or reject.

Our main result is the following. A collection of  $n$ -bit strings  $x_1, \dots, x_t$  are stored at  $t$  terminal nodes  $u_1, \dots, u_t$  in a network  $G = (V, E)$ , where node  $u_i$  stores  $x_i$ . We denote  $\text{EQ}_n^t$  the problem of checking the equality  $x_1 = \dots = x_t$  between the  $t$  strings. Let us define the *radius* of a given instance of  $\text{EQ}_n^t$  as  $r = \min_i \max_j \text{dist}_G(u_i, u_j)$ , where  $\text{dist}_G$  denotes the distance in the (unweighted) graph  $G$ . Our main result is the design of a dQMA protocol for  $\text{EQ}_n^t$ , using small certificate. This can be summarized by the following informal statement (the formal statement is in Section 5):

► **Main Result.** *There is a distributed Quantum Merlin-Arthur (dQMA) protocol for certifying equality between  $t$  binary strings ( $\text{EQ}_n^t$ ) of length  $n$ , and located at a radius- $r$  set of  $t$  terminals, in a single round of communication between neighboring nodes using certificates of size  $O(tr^2 \log n)$  qubits, and messages of size  $O(tr^2 \log(n + r))$  qubits.*

It is worth mentioning that, although the dependence in  $r$  and  $t$  is polynomial, the dependence in the actual size  $n$  of the instance remains logarithmic, which is our main concern. Indeed, for applications such as the aforementioned distributed data storage motivating the distant  $\text{EQ}_n^t$  problem, it is expected that both the number  $t$  of replicas, and the maximum distance between the nodes storing these replicas are of several orders of magnitude smaller than the size  $n$  of the stored replicated data.

It is also important to note that our protocol satisfies the basic requirement of *reusability*, as one aims for protocols enabling regular and frequent verifications that the data are not corrupted. Specifically, the quantum operations performed on the certificates during the local verification phase operated between neighboring nodes preserve the quantum nature of these certificates. That is, if  $\text{EQ}_n^t$  is satisfied, i.e., if all the replicas  $x_i$ 's are equal, then, up to an elementary local relocation of the quantum certificates, these certificates are available for a next test. If  $\text{EQ}_n^t$  is not satisfied, i.e., if there exists a pair of replicas  $x_i \neq x_j$ , then the certificates do not need to be preserved as this scenario corresponds to the case where the correctness of the data structure is violated, requiring the activation of recovery procedures for fixing the bug, and reassigning certificates to the nodes.

Our quantum protocol is based on the SWAP test [12], which is a basic tool in the theory of quantum computation and quantum information. This test allows to check if a quantum state is symmetric, and has several applications, such as estimating the inner product of two states (e.g., [12, 9, 47]), checking whether a given state (or a reduced state of it) is pure or entangled with the environment system (e.g., [1, 33, 25, 32]), and more. In this paper, we use the SWAP test in yet another way: *for checking if two of the reduced states of a given state are close*. A similar use was done by Rosgen [44] in a different context – transforming quantum circuits to shallow ones in a hardness reduction proof.

Finally, observe that our logarithmic upper bound for dQMA protocols is in contrast to the linear lower bound that can be shown for classical dMA protocols even for  $t = 2$  on a path of 4 nodes and even for the case where communication between the neighboring nodes is extended to multiple rounds (see precise statement and proof in Section 6). Our results thus show that quantum certification mechanism can provide an exponential advantage over classical certification mechanisms.

## 1.2 Related Work

The concept of distributed proofs is a part of the framework of distributed network computing since the early works on fault-tolerance (see, e.g., [2, 6, 26]). Proof-labeling schemes were introduced in [35], and variants have been studied in [24, 20]. Randomized proof-labeling schemes have been studied in [22]. Extensions of distributed proofs to a hierarchy of decision mechanisms have been studied in [18] and [7]. Frameworks like cloud computing recently enabled envisioning systems in which the nodes of the network could interact with a third party, leading to the concept of *distributed interactive proofs* [34]. There, each node can interact with an *oracle* who has a complete view of the system, is computationally unbounded, but is not trustable. For instance, in Arthur-Merlin (dAM) protocols, the nodes start by querying the oracle Merlin, which provides them with answers in their certificates. There is a simple classical compact dAM protocol for distant EQ, where the two players stand at the extremities of a path (see Section 3). We refer to [15, 21, 39] for recent developments in the framework of distributed interactive proofs. While distributed Arthur-Merlin protocols and their extensions provide an appealing theoretical framework for studying the power of interactive proofs in the distributed setting, the practical implementation of such protocols remains questionable, since they all require the existence of a know-all oracle, Merlin, and it is unclear if a Cloud could play this role. On the other hand, in dMA and dQMA protocols, interaction with an external party is not required, but only a one-time assignment of certificates is needed, which are then reusable for regular verification. As in the classical proof-labeling schemes setting, these certificates can actually be *created* by the nodes themselves during a pre-processing phase, making the reliance on a know-all oracle unnecessary.

After a few early works [8, 17, 23, 45] that shed light on the potential and limitations of quantum distributed computing (see also [5, 11, 16] for general discussions), evidence of the advantage of quantum distributed computing over classical distributed computing have been obtained recently for three fundamental models of (synchronous fault-free) distributed network computing: the CONGEST model [28, 37], the CONGEST-CLIQUE model [27] and the LOCAL model [38]. The present paper adds to this list another important task for which quantum distributed computing significantly outperforms classical distributed computing, namely, distributed certification.

Note that while this paper is the first to study quantum Merlin-Arthur protocols in a distributed computing framework, there are a number of prior works studying them in communication complexity [43, 30, 31, 10]. In particular, quantum Merlin-Arthur protocols

are shown to improve some computational measure (say, the total length of the messages from the prover to Alice, and of the messages between Alice and Bob) exponentially compared to Merlin-Arthur protocols where the messages from the prover are classical [43, 31].

The question of computing functions on inputs that are given to graph nodes was also studied in the context of communication complexity. The equality function was studied for the case where all nodes have inputs [4]. Other works considered a setting similar to ours, i.e., where only some nodes have inputs [13, 14], but did not study the equality problem.

## 2 Model and Definitions

**Distributed verification on graphs.** Let  $t \geq 2$ , and let  $f: (\{0, 1\}^n)^t \rightarrow \{0, 1\}$  be a function. The aim of the nodes is to collectively decide whether  $f(x_1, \dots, x_t) = 1$  or not, where  $x_1, \dots, x_t$  are assigned to  $t$  nodes of a graph. Specifically, an instance of the problem  $f$  is a  $t$ -tuple  $(x_1, \dots, x_t) \in \{0, 1\}^n \times \dots \times \{0, 1\}^n$ , a connected graph  $G = (V, E)$ , and an ordered sequence  $v_1, \dots, v_t$  of distinct nodes of  $G$ . The node  $v_i$  is given  $x_i$  as input, for  $i = 1, \dots, t$ . All the other nodes receive no inputs. We consider distributed Merlin-Arthur (dMA) protocols for deciding whether  $f(x_1, \dots, x_t) = 1$ , in which a non-trustable *prover* (Merlin) assigns (or “sends”) *certificates* to the nodes, and then the nodes (Arthur) perform a 1-round randomized verification algorithm. The verification algorithm consists of each node simultaneously sending messages to all its immediate neighbors, receiving messages from them, then performing a local computation, and finally accepting or rejecting locally.<sup>1</sup> We say that a dMA protocol has *completeness*  $a$  and *soundness*  $b$  for a function  $f$  if the following holds for every  $(x_1, \dots, x_t) \in \{0, 1\}^n \times \dots \times \{0, 1\}^n$ , every connected graph  $G$ , and every ordered sequence  $v_1, \dots, v_t$  of distinct nodes in  $G$ :

- (**completeness**) if  $f(x_1, \dots, x_t) = 1$ , then the prover can assign certificates to the nodes such that  $\Pr[\text{all nodes accept}] \geq a$ ;
- (**soundness**) if  $f(x_1, \dots, x_t) = 0$ , then, for every certificate assignment by the prover,  $\Pr[\text{all nodes accept}] \leq b$ .

The completeness condition guarantees that, when the system is in a “legal” state (specified by  $f(x_1, \dots, x_t) = 1$ ), with probability at least  $a$  all nodes accept. The soundness condition guarantees that, when the system is in an “illegal” state (specified by  $f(x_1, \dots, x_t) = 0$ ), with probability at least  $1 - b$  at least one node rejects. The value  $b$  represents the error probability of the protocol on an illegal instance, and thus we sometimes refer to it as the *soundness error*. A node detecting illegality of the state can raise an alarm, or launch a recovery procedure. Protocols with completeness 1 are called 1-sided protocols, or protocols with perfect completeness. Similarly to prior works on distributed verification, the certificate size of the protocol is measured as the maximum size (over all the nodes of the network) of the certificate sent by the prover to one of the nodes, and the message size of the protocol is measured as the maximum size (over all pairs of adjacent nodes) of the message exchanged between two adjacent nodes. Specifically, we will consider the multi-party version of the equality function,  $\text{EQ}_n^t$ , which is the boolean-valued function from  $(\{0, 1\}^n)^t$  such that  $\text{EQ}_n^t(x_1, \dots, x_t) = 1 \iff x_1 = \dots = x_t$ .

<sup>1</sup> We can naturally extend this definition to define dMA protocols with  $\mu$  rounds of communication among neighbors, for any integer  $\mu \geq 1$ . In this paper, however, we focus on the case  $\mu = 1$  since all the protocols we design use only 1-round verification algorithms. The only exception is Section 6, where we show classical lower bounds that hold even for  $\mu > 1$ .



In this work, we extend the framework of dMA protocols, to consider also cases where the certificates given to the nodes can contain qubits (although they may also contain classical bits) and the nodes can exchange messages consisting of qubits. These will be called *distributed Quantum Merlin-Arthur* (dQMA) protocols. More precisely, in a dQMA protocol for a function  $f$ , a non-trustable prover first sends a certificate to each node, which consists of a quantum state and classical bits; the quantum states may be entangled, even though all our quantum protocols do not require any prior entanglement, nor any shared classical random bits. Then the nodes perform a 1-round quantum verification algorithm, where each node simultaneously sends a quantum message to all its immediate neighbors, receives quantum messages from them, then performs a local computation, and finally accepts or rejects locally. Note that, as opposed to the classical setting, we cannot assume that a node simply broadcasts its certificate to all its neighbors, as quantum states cannot be duplicated. However, a node can still send copies of the classical parts of the certificate. We define completeness and soundness of dQMA protocols as for dMA protocols.

► **Remark.** A special case of interest is when the graph  $G$  is a path  $v_0, \dots, v_r$ ,  $r \geq 1$ , where the left-end node  $v_0$  has an  $n$ -bit string  $x$  as input, the right-end node  $v_r$  has an  $n$ -bit string  $y$  as input, and the intermediate nodes  $v_1, \dots, v_{r-1}$  have no inputs. That is,  $t = 2$ . Given a function  $f: \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$ , the aim of the nodes is to collectively decide whether  $f(x, y) = 1$  or not. This setting is very much related to communication complexity.

**Classical two-party communication complexity.** We refer to [36] for the basic concepts of two-party communication complexity. In this paper we will only consider two-party one-way communication complexity. In this model two parties, denoted Alice and Bob, each receives an input  $x \in \{0, 1\}^n$  and  $y \in \{0, 1\}^n$ , respectively. The goal is for Bob to output the value  $f(x, y)$  for some known Boolean function  $f: \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$ . Only Alice can send a message to Bob. The one-way two-sided-error communication complexity of  $f$  is the minimum number of bits that have to be sent on the worst input in a protocol that outputs the correct answer with probability at least  $2/3$ . The one-way one-sided-error communication complexity of  $f$  is the minimum number of bits that have to be sent on the worst input in a protocol that outputs the correct answer with probability 1 on any 1-input, and outputs the correct answer with probability at least  $2/3$  on any 0-input.

We shall especially consider the following two functions. The equality function  $\text{EQ}_n$  is defined as  $\text{EQ}_n(x, y) = 1$  when  $x = y$  and  $\text{EQ}_n(x, y) = 0$  otherwise, for any  $x, y \in \{0, 1\}^n$ . Its one-way one-sided-error communication complexity is  $O(\log n)$  – see, e.g., [36]. For any integer  $d \geq 0$ , the Hamming distance function  $\text{HAM}_n^d$  is defined as follows: for any  $x, y \in \{0, 1\}^n$ ,  $\text{HAM}_n^d(x, y) = 1$  if the Hamming distance between  $x$  and  $y$  is at most  $d$ , and  $\text{HAM}_n^d(x, y) = 0$  otherwise. It is known [47] that, for  $d$  constant, the one-way two-sided-error communication complexity of  $\text{HAM}_n^d$  is  $O(\log n)$ .

For any Boolean function  $f: \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$ , a set  $S \subseteq \{0, 1\}^n \times \{0, 1\}^n$  is a *1-fooling set* for  $f$  if, on the one hand, for every  $(x, y) \in S$ ,  $f(x, y) = 1$ , and, on the other hand, for every two pairs  $(x_1, y_1) \neq (x_2, y_2)$  in  $S \times S$ ,  $f(x_1, y_2) = 0$  or  $f(x_2, y_1) = 0$ .

**Quantum two-party communication complexity.** We assume the reader is familiar with the basics of quantum computation, in particular the notion of qubits, Dirac notation such as  $|\psi\rangle$  and  $\langle\psi| := (|\psi\rangle)^\dagger$ , and the quantum circuit model (see Sections 2 and 4 in Ref. [40], for instance). In the full version of this paper, we present more advanced concepts such as mixed states that will be used in some of our proofs.

Quantum two-party communication complexity, first introduced by Yao [46], is defined

similarly to the classical version. The only difference is that the players are allowed to exchange qubits instead of bits (the cost of a quantum protocol is the number of qubits sent by the protocol). Note that since quantum protocols can trivially simulate classical protocols, the quantum communication complexity of a function is never larger than its classical communication complexity. More precisely, an  $m$ -qubit one-way quantum protocol  $\pi$  for the function  $f$  can be described in its most general form as follows. Alice prepares an  $m$ -qubit (pure) quantum state  $|h_x\rangle$  and sends it to Bob.<sup>2</sup> Bob then makes a measurement on the state  $|h_x\rangle$ , which gives an outcome  $b \in \{0, 1\}$ . Finally, Bob outputs  $b$ . Since Bob's measurement in the above description depends only on his input  $y$ , it can be mathematically described, for each  $y \in \{0, 1\}^n$ , by two positive semi-definite matrices  $M_{y,0}$  and  $M_{y,1}$  such that  $M_{y,0} + M_{y,1} = I$ . This pair  $\{M_{y,0}, M_{y,1}\}$  is called a POVM measurement (POVM measurements are the most general form of measurements allowed by quantum mechanics). If  $|h_x\rangle$  is measured by the POVM  $\{M_{y,0}, M_{y,1}\}$ , the probability that  $b = 0$  is  $\text{tr}(M_{y,0}(|h_x\rangle\langle h_x|))$ , while the probability that  $b = 1$  is  $\text{tr}(M_{y,1}(|h_x\rangle\langle h_x|))$ .

### 3 General Overview of our Techniques

Let us provide an intuition of our protocol in the case of  $\text{EQ}_n^2$  over a path  $v_0, \dots, v_r$  of length  $r \geq 1$  in which the terminals are the two nodes  $v_0$  and  $v_r$  (that we rename Alice and Bob, for convenience). Let us call  $x$  and  $y$  the  $n$ -bit strings owned by Alice and Bob, respectively. There is a simple classical protocol for distant equality in a somewhat similar setting, where the verifier (Arthur, consisting of all the graph nodes) can send random bits to the prover (Merlin) before receiving the certificates; this is called a **dAM** protocol. In this protocol, Alice picks a hash function  $h$  at random in an appropriate family of hash functions (i.e., a family such that both  $h$  and  $h(x)$  can be encoded using  $O(\log n)$  bits and such that the probability that  $h(x) \neq h(y)$  is high when  $x \neq y$ ). Merlin provides every node with the certificate  $(h, h(x))$ , each node checks it received the same certificates as its neighbors, and Bob additionally checks whether  $h(x) = h(y)$ . Obviously, one cannot switch the order of Arthur and Merlin, as letting Merlin choose the hash function would enable him to fool Arthur on illegal instances by picking  $h$  that hashes identically the distinct input strings  $x$  and  $y$ . The main idea of our **dQMA** protocol is to ask Merlin to provide the nodes with a quantum certificate consisting of the *quantum superposition* of all the possible hashes.

Entering slightly more into the details, for any  $x \in \{0, 1\}^n$  we consider the *quantum fingerprint*  $|h_x\rangle = \frac{1}{\sqrt{K}} \sum_h |h\rangle |h(x)\rangle$ , where the sum is over all the hash functions, and  $\frac{1}{\sqrt{K}}$  is the normalization factor of the quantum state. By using the same family of hash functions as in the aforementioned **dAM** protocol, these fingerprints can be constructed in such a way that their length is  $O(\log n)$  qubits, and  $|h_x\rangle$  and  $|h_y\rangle$  are very far (more precisely, almost orthogonal) when  $x \neq y$ . Checking whether the two quantum fingerprints  $|h_x\rangle$  and  $|h_y\rangle$  are either equal or far apart can be achieved by a quantum test called the **SWAP** test [12]. Formally, the probability that the **SWAP** test accepts is  $1/2 + |\langle h_x | h_y \rangle|^2 / 2$ , where  $\langle h_x | h_y \rangle$  denotes the inner product between the two quantum states  $|h_x\rangle$  and  $|h_y\rangle$ .

Let us now describe the outline of our **dQMA** protocol. In the protocol each intermediate node  $v_1, \dots, v_{r-1}$  expects to receive the quantum fingerprint  $|h_x\rangle$ . Alice, who does not receive any certificate, creates by herself the fingerprint  $|h_x\rangle$ , which depends only on  $x$ . Similarly, Bob creates by himself the fingerprint  $|h_y\rangle$ . The checking procedure simply checks whether

<sup>2</sup> Without loss of generality, we assume that Alice does not use any mixed state (i.e., a probability distribution on pure states) in her message, as she can simulate it using a pure state called the *purification* [40] whose length is at most twice the one of the mixed state.

all these  $(r + 1)$  fingerprints are equal. This is done by applying the SWAP test to check whether the fingerprints owned by adjacent nodes are equal or not. There are however a few subtleties. In particular, since our analysis crucially requires that the SWAP tests do not overlap, for each node we need to decide whether it will perform the SWAP test with its right neighbor or its left neighbor. We do it in a randomized way and deal carefully with the conflicting choices that appear. For the case  $x = y$  all the SWAP tests then succeed with probability 1 and thus all the nodes accept.

For the case  $x \neq y$ , let us provide some intuition about why a prover cannot fool the nodes for convincing them to all accept. To simplify the description we assume below that  $|h_x\rangle$  and  $|h_y\rangle$  are orthogonal (instead of only almost orthogonal). If the prover was forced to send certificates restricted to *product states* of the form  $|g_1\rangle \otimes |g_2\rangle \otimes \cdots |g_{r-1}\rangle$  where  $|g_j\rangle$  is the state to the  $j$ th node, then a fairly straightforward argument would guarantee that, with large probability, at least one node rejects. Indeed, under the product states restriction, intuitively the best strategy for the prover to cheat is to send states “intermediate” between  $|h_x\rangle$  and  $|h_y\rangle$ , namely, to send the state  $|g_j\rangle = \cos(\pi j/2r)|h_x\rangle + \sin(\pi j/2r)|h_y\rangle$  to node  $v_j$  for each  $j \in \{1, \dots, r-1\}$ . Then, the probability that all nodes accept when performing the SWAP tests would be roughly  $\prod_{j=1}^{r-1} (1/2 + |\langle g_j | g_{j+1} \rangle|^2/2) = 1 - \Omega(1/r)$ . The cheating prover could then be caught with probability  $\Omega(1/r)$ , and this probability can be amplified to  $\Omega(1)$  by asking the prover to send several copies of the certificates (amplification is possible since our protocol has perfect completeness).

The formal analysis of the protocol however faces several difficulties, which are mostly due to the nature of quantum computation, and are especially challenging to handle in the framework of distributed computation. For instance, quantum states cannot be duplicated (the “no-cloning Theorem”), which implies that a same quantum state cannot be used for parallel tests. Additionally, even sequential tests face the difficulty that the first test may collapse the quantum state, making the second test impossible to perform (or at least significantly complicating the analysis of the second test). Thus node  $v_i$  cannot perform the SWAP test with its two neighbors  $v_{i-1}$  and  $v_{i+1}$  simultaneously and (as already mentioned) we have to design carefully the protocol so that the SWAP tests do not overlap. A second, and much more problematic issue is that the non-trustable Merlin can send arbitrary certificates to the nodes for fooling them. In particular it is not restricted to send certificates that are product states. A priori, it may seem that the SWAP test is not strong enough to handle fooling strategies beyond product states. In this work we show that the SWAP test can actually detect such fooling strategies.

Specifically, our approach consists in considering the so-called *reduced states*, and to establish the following property of the SWAP test (cf. Lemma 5 in Section 4). If the SWAP test accepts with high probability when applied on the part of any two adjacent nodes in a (possibly non-product) global quantum state resulting from the certificates, then the two reduced states of that part (which is a bipartite state) must be close. As the two states  $|h_x\rangle$  and  $|h_y\rangle$  are very far apart when  $x \neq y$ , we can thus use this result to show that there is a good probability that the SWAP test rejects at some node. Moreover, using reduced states allows us to overcome other technical difficulties in the analysis of the (non-overlapping) SWAP tests we consider. Indeed, some form of average-case success probability of all the SWAP tests can be considered, instead of having to argue about the probability that all the SWAP tests individually accept.

## 4 Quantum Distributed Proofs on Paths

In this section we restrict ourselves to the case of a path  $v_0, \dots, v_r$  of length  $r \geq 1$ , in which only the two extremities  $v_0$  and  $v_r$  are given inputs. This framework allows us to elaborate our main technique, that will be extended to arbitrary graphs in Section 5. Let  $x \in \{0, 1\}^n$  be the input to  $v_0$ , and  $y \in \{0, 1\}^n$  be the input to  $v_r$ . Our goal is to design a dQMA protocol to decide whether  $f(x, y) = 1$  or not, for some given Boolean function on  $\{0, 1\}^n \times \{0, 1\}^n$ .

We show the following general theorem that converts a one-way quantum communication complexity protocol into a quantum Merlin-Arthur protocol for the corresponding long-distance problem on the path. This theorem applies not only to one-sided-error protocols, but also to the two-sided-error case (with a logarithmic additional factor in the complexity).

► **Theorem 1.** *Let  $f: \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$  be a Boolean function.*

- *If  $f$  has a quantum one-way one-sided-error communication protocol transmitting at most  $q$  qubits, then there exists a 1-sided distributed quantum Merlin-Arthur protocol for  $f$  on the path of length  $r$ , with soundness  $1/3$ , using certificates of size  $O(r^2 q)$  qubits, and exchanging messages of length  $O(r^2(q + \log r))$  qubits.*
- *If  $f$  has a quantum one-way two-sided-error communication protocol transmitting at most  $q$  qubits, then, for any constant  $c$ , there exists a distributed quantum Merlin-Arthur protocol for  $f$  on the path of length  $r$  with completeness  $1 - 1/n^c$ , soundness  $1/3$ , using certificates of size  $O(r^2 q \log(n + r))$  qubits, and exchanging messages of length  $O(r^2 q \log(n + r))$  qubits.*

Using known results (cf. Section 2) about one-way communication complexities of  $\text{EQ}_n$  and  $\text{HAM}_n^d$ , the following two results are direct applications of Theorem 1.

► **Corollary 2.** *There exists a one-sided quantum Merlin-Arthur protocol for  $\text{EQ}_n$  in the path of length  $r$  with soundness  $1/3$ , using certificates of size  $O(r^2 \log n)$  qubits, and exchanging messages of length  $O(r^2 \log(n + r))$  qubits.<sup>3</sup>*

► **Corollary 3.** *For any  $c > 0$  and  $d > 0$ , there exists a quantum Merlin-Arthur protocol for  $\text{HAM}_n^d$  in the path of length  $r$  with completeness  $1 - 1/n^c$ , soundness  $1/3$ , using certificates of length  $O(r^2(\log n) \log(n + r))$  qubits, and exchanging messages of length  $O(r^2(\log n) \log(n + r))$  qubits.*

The rest of this section is dedicated to proving Theorem 1. Let us first give an overview of the proof. In our dQMA protocol in the path, the verification algorithm performed by the nodes on the line is merely a simulation of a two-party one-way quantum communication complexity protocol  $\pi$  between Alice and Bob for the function  $f(x, y)$ , with the help of certificates provided by the prover. Specifically, every intermediate node  $v_1, \dots, v_{r-1}$  expects to receive the quantum state sent by Alice to Bob in  $\pi$ , as certificate. Let us denote by  $|h_x\rangle$  this state, which depends on  $x$ . The right-end node  $v_r$  simulates the two-party protocol  $\pi$  using  $|h_x\rangle$  received from the left neighbor  $v_{r-1}$ , and applying Bob's measurement (i.e., the POVM measurement). If  $f(x, y) = 1$ , the prover honestly sends the desired state, and  $v_r$  accepts as it does receive  $|h_x\rangle$ . However, if  $f(x, y) = 0$ , then the malicious prover does not necessarily send a desired state. To catch the potentially malicious behavior of the prover on “illegal” instances (i.e., those for which  $f(x, y) = 0$ ), each intermediate node checks whether its local proof is “close to” the one of its right neighbor. This is performed by an application of the SWAP test.

<sup>3</sup> Here we are using the fact that  $\log n + \log r$  is of the same order as  $\log(n + r)$  for conciseness.

Section 4.1 below describes in more detail how to construct the distributed quantum Merlin-Arthur protocol, denoted  $\mathcal{P}_\pi$ , from an arbitrary one-way quantum communication protocol  $\pi$  for the function  $f$ . Section 4.2 analyzes the completeness and the soundness of the protocol  $\mathcal{P}_\pi$ . Finally, Section 4.3 shows how to reduce the soundness error using “parallel repetitions” and how to apply this analysis to prove Theorem 1.

#### 4.1 A dQMA Protocol for the Path

Let  $\varepsilon \geq 0$  be a constant, which will be fixed small enough later in the proof. Let  $\pi$  be a quantum one-way communication protocol for  $f$  transmitting at most  $q$  qubits, such that, for every input pair  $(x, y)$ , if  $f(x, y) = 1$  then  $\pi$  outputs 1 with probability at least  $1 - \varepsilon$ , and if  $f(x, y) = 0$  then  $\pi$  outputs 0 with probability at least  $2/3$ . Let  $|h_x\rangle$  be the  $q$ -qubit (pure) state sent from Alice to Bob, and let  $\{M_{y,1}, M_{y,0}\}$  be the POVM measurement performed by Bob on  $|h_x\rangle$ , where  $M_{y,1}$  corresponds to the measurement result 1 (accept) and  $M_{y,0}$  to the measurement result 0 (reject). Our quantum Merlin-Arthur protocol  $\mathcal{P}_\pi$  is as follows.

**Protocol  $\mathcal{P}_\pi$  for function  $f$  on input pair  $(x, y)$  in path  $v_0, \dots, v_r$ :**

1. If  $f(x, y) = 1$  then the prover sends the quantum register  $R_j$  that has the state  $|h_x\rangle$  (or  $|h_x\rangle\langle h_x|$  as the mixed state representation) as certificate to each of the intermediate nodes  $v_j$ ,  $j \in \{1, \dots, r-1\}$ .
2. The left-end node  $v_0$  prepares the state  $\rho_0 = |h_x\rangle\langle h_x|$  in quantum register  $R_0$ .
3. For every  $j = 0, \dots, r-1$ , the node  $v_j$  chooses a bit  $b_j$  uniformly at random, and sends its quantum register  $R_j$  to the right neighbor  $v_{j+1}$  whenever  $b_j = 0$ .
4. For every  $j = 1, \dots, r-1$ , if  $v_j$  receives a quantum register from its left neighbor  $v_{j-1}$ , and if  $b_j = 1$ , then  $v_j$  performs the SWAP test on the registers  $(R_{j-1}, R_j)$ , and accepts or rejects accordingly; Otherwise,  $v_j$  accepts.
5. If the right-end node  $v_r$  receives a quantum register  $R_{r-1}$  from its left neighbor, then  $v_r$  performs the POVM measurement  $\{M_{y,1}, M_{y,0}\}$  corresponding to  $\pi$  applied to the state in  $R_{r-1}$ , and accepts or rejects accordingly; Otherwise,  $v_r$  accepts.

In the above protocol  $\mathcal{P}_\pi$ , the size of the quantum certificate that each node receives from the prover is at most  $q$ , and the length of the quantum message that each node sends to the neighbor is also at most  $q$ . In the next subsection, we prove that the above protocol has completeness  $1 - \varepsilon/2$  and soundness  $1 - 1/42r^2$ .

#### 4.2 Analysis of Protocol $\mathcal{P}_\pi$

For the analysis, we recall the SWAP test. The test is a protocol with a given input state on  $\mathcal{H} = \mathcal{H}_1 \otimes \mathcal{H}_2$ , where  $\mathcal{H}_1$  and  $\mathcal{H}_2$  are complex Euclidian spaces. Here, we consider  $\mathcal{H}_1$  and  $\mathcal{H}_2$  as quantum registers  $R_1$  and  $R_2$ .

**SWAP test** on a pure state  $|\psi\rangle$  on  $\mathcal{H}$ , which is given in registers  $(R_1, R_2)$ .

1. Prepare the single-qubit state  $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$  in register  $R_0$ .
2. If the content of  $R_0$  is 1, then apply the swap operator  $S$  on the state  $|\psi\rangle$  in registers  $(R_1, R_2)$ , where  $S$  is defined by  $S(|j_1\rangle|j_2\rangle) = |j_2\rangle|j_1\rangle$  (namely,  $S$  swaps register  $R_1$  and register  $R_2$ ).
3. Apply the Hadamard operator  $H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$  on the state in register  $R_0$ , and measure the content in the standard basis. Accept if the content is 0, and reject otherwise.

**Completeness.** The following lemma is a direct consequence of the definition of the SWAP test. Here,  $\mathcal{H}_S$  is the symmetric subspace in  $\mathcal{H}$  (namely, the subspace spanned by the states invariant by the swap operator  $S$ , or equivalently, the eigenstates of  $S$  with eigenvalue 1), and  $\mathcal{H}_A$  is the anti-symmetric subspace in  $\mathcal{H}$  (namely, the subspace spanned by the eigenstates of  $S$  with eigenvalue  $-1$ ). Note that any state in  $\mathcal{H}$  is represented as the superposition of a state in  $\mathcal{H}_S$  (symmetric state) and a state in  $\mathcal{H}_A$  (anti-symmetric state) as the swap operator  $S$  is a Hermitian matrix that only has  $+1$  and  $-1$  eigenvalues.

► **Lemma 4.** *Assume that  $|\psi\rangle = \alpha|\psi_S\rangle + \beta|\psi_A\rangle$  where  $|\psi_S\rangle \in \mathcal{H}_S$  and  $|\psi_A\rangle \in \mathcal{H}_A$ . Then, the SWAP test on input  $|\psi\rangle$  accepts with probability  $|\alpha|^2$ .*

For the completeness, assume  $f(x, y) = 1$ . The prover then sends  $|h_x\rangle$  to all the intermediate nodes. Then, all the nodes except the right-end node have  $|h_x\rangle$ . By Lemma 4, all the SWAP tests done in Step 4 are accepted with probability 1 (note that  $|h_x\rangle \otimes |h_x\rangle$  is a symmetric state). Furthermore, the right-end node accepts with probability at least  $(1 - \varepsilon)/2 + 1/2 = 1 - \varepsilon/2$  as  $v_r$  can receive  $|h_x\rangle$  and simulate  $\pi$  with probability  $1/2$  and accepts otherwise in Step 5.

**Soundness.** The following lemma presents a crucial property of the SWAP test: its applicability for checking whether the two *reduced* states are close. It is a trace-distance version of a lemma by Rosgen [44, Lemma 5.1]. Here, a reduced state intuitively represents the local information on its own quantum system, by disregarding the outside systems. Note that the trace distance between two quantum states  $\rho$  and  $\sigma$  is characterized as  $\text{dist}(\rho, \sigma) = \max_M \text{tr}(M(\rho - \sigma))$ , where the maximization is taken over all positive semi-definite matrices  $M$  such that  $M \leq I$ .

► **Lemma 5.** *Let  $z \geq 1$ , and assume that the SWAP test on input  $\rho$  in the input register  $(R_1, R_2)$  accepts with probability  $1 - 1/z$ . Then,  $\text{dist}(\rho_1, \rho_2) \leq 2/\sqrt{z} + 1/z$ , where  $\rho_j$  is the reduced state on  $R_j$  of  $\rho$ . Moreover, if the SWAP test on input  $\rho$  accepts with probability 1, then  $\rho_1 = \rho_2$  (and hence  $\text{dist}(\rho_1, \rho_2) = 0$ ).*

For the soundness, let  $(x, y)$  be any pair such that  $f(x, y) = 0$ .

► **Lemma 6.** *For every  $j \in \{1, \dots, r\}$ , let  $F_j$  be the event that  $v_j$  performs the local test (SWAP or POVM) in Protocol  $\mathcal{P}_\pi$ , and let  $E_j$  be the event that this local test rejects. Then we have  $\sum_{j=1}^r \Pr[E_j | F_j] \geq \frac{1}{2^{1/r}}$ .*

**Proof.** Let  $\alpha_j = \Pr[E_j | F_j]$ . Then, for every  $j \in \{1, \dots, r\}$ ,  $\Pr[\overline{E_j} | F_j] = 1 - \alpha_j$ , where we note that the complementary event  $\overline{E_j}$  for  $j = 1, \dots, r-1$  is the event where the SWAP test on the two  $q$ -qubit states  $\rho_{j-1}$  and  $\rho_j$  accepts, and  $\overline{E_r}$  represents the event that the result of the POVM measurement is 1 (accept), which corresponds to the POVM element  $M_{y,1}$ . By Lemma 5, the trace distance  $\text{dist}$  between the reduced  $q$ -qubit states  $\rho_{j-1}$  on  $v_{j-1}$  and  $\rho_j$  on  $v_j$  is

$$\text{dist}(\rho_{j-1}, \rho_j) \leq \begin{cases} \frac{2}{\sqrt{1/\alpha_j}} + \frac{1}{1/\alpha_j} & \text{if } \alpha_j \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

and thus  $\text{dist}(\rho_{j-1}, \rho_j) \leq 3\sqrt{\alpha_j}$ . Then, by the triangle inequality,

$$\text{dist}(\rho_0, \rho_{r-1}) \leq \sum_{j=1}^{r-1} \text{dist}(\rho_{j-1}, \rho_j) \leq 3 \sum_{j=1}^{r-1} \sqrt{\alpha_j}.$$

## 28:12 Distributed Quantum Proofs for Replicated Data

For  $\Pr[E_r|F_r]$ , the soundness of  $\pi$  promises that the probability that the test  $\{M_{y,1}, M_{y,0}\}$  rejects  $\rho_0 = |h_x\rangle\langle h_x|$  is at least  $2/3$ , i.e.,  $\text{tr}(M_{y,0}\rho_0) \geq 2/3$ . Hence,

$$\alpha_r = \Pr[E_r|F_r] = \text{tr}(M_{y,0}\rho_{r-1}) \geq \frac{2}{3} - \text{dist}(\rho_0, \rho_{r-1}) \geq \frac{2}{3} - 3 \sum_{j=1}^{r-1} \sqrt{\alpha_j},$$

where the first inequality comes from the characterization of  $\text{dist}$  on indistinguishability of two states, that is,  $|\text{tr}(M_{y,0}\rho_0) - \text{tr}(M_{y,0}\rho_{r-1})| \leq \text{dist}(\rho_0, \rho_{r-1})$ . Thus, we have

$$3 \sum_{j=1}^r \sqrt{\alpha_j} \geq \alpha_r + 3 \sum_{j=1}^{r-1} \sqrt{\alpha_j} \geq \frac{2}{3}.$$

By the Cauchy-Schwarz inequality,

$$\sqrt{r} \sqrt{\sum_{j=1}^r \alpha_j} \geq \sum_{j=1}^r \sqrt{\alpha_j},$$

and thus we have

$$\sum_{j=1}^r \alpha_j \geq \left( \frac{2}{9\sqrt{r}} \right)^2 \geq \frac{1}{21r}. \quad \blacktriangleleft$$

In Steps 4 and 5, node  $v_j$ ,  $1 \leq j \leq r$ , performs the local test (SWAP or POVM) with probability at least  $1/4$  (more precisely,  $v_j$ ,  $1 \leq j \leq r-1$ , performs the SWAP test with probability  $1/4$  and  $v_r$  performs the POVM with probability  $1/2$ ). It follows that, for every  $j \in \{1, \dots, r\}$ , the event  $F_j$  occurs in at least  $(1/4) \times 2^r$  outcomes of all the  $2^r$  possible outcomes  $b_0 \cdots b_{r-1}$  that can be obtained in Step 3. Here, we consider any fixed outcomes  $b_0 \cdots b_{r-1}$  that induce  $k$  events  $F_{j_1}, F_{j_2}, \dots, F_{j_k}$  with  $k \neq 0$  where we note that  $0 \leq k \leq \lfloor r/2 \rfloor$  in general. The probability that some node rejects in Steps 4 or 5 *under this outcome* is

$$\Pr[\vee_{t=1}^k E_{j_t} | \wedge_{i'=1}^k F_{j_{i'}}] \geq \frac{1}{\lfloor r/2 \rfloor} \sum_{i=1}^k \Pr[E_{j_i} | \wedge_{i'=1}^k F_{j_{i'}}] = \frac{1}{\lfloor r/2 \rfloor} \sum_{i=1}^k \Pr[E_{j_i} | F_{j_i}],$$

where the inequality follows from  $k \Pr[\vee_{t=1}^k E_{j_t}] = \sum_{i=1}^k \Pr[\vee_{t=1}^k E_{j_t}] \geq \sum_{i=1}^k \Pr[E_{j_i}]$  and  $k \leq \lfloor r/2 \rfloor$ , and the equality comes from the fact that each  $F_{j_i}$  and  $E_{j_i}$  are independent from all the other event  $F_{j_{i'}}$  with  $i' \neq i$  (note that  $|j_{i'} - j_i| \geq 2$  since  $F_{j-1}$  and  $F_j$  never occur at the same time). As each outcome occurs with probability  $1/2^r$ , the probability that some node rejects in Steps 4 or 5 is at least

$$\frac{1}{2^r} \cdot [(1/4) \cdot 2^r] \cdot \frac{1}{\lfloor r/2 \rfloor} \sum_{j=1}^r \Pr[E_j | F_j] \geq \frac{1}{2^r} \sum_{j=1}^r \Pr[E_j | F_j] \geq \frac{1}{2^r} \cdot \frac{1}{21r} = \frac{1}{42r^2},$$

where the second last inequality comes from Lemma 6.

### 4.3 Proof of Theorem 1

So far, we have shown that the protocol  $\mathcal{P}_\pi$  has a completeness parameter very close to 1, but high soundness error. To establish Theorem 1, we need to reduce the soundness error without degrading the completeness too much. This is achieved via a form of parallel repetition of  $\mathcal{P}_\pi$ , by taking the logical conjunction of the outputs obtained by repetitions. The protocol resulting from  $k$  repetitions of  $\mathcal{P}_\pi$  is denoted by  $\mathcal{P}_\pi[k]$ , and works as follows.



**Protocol  $\mathcal{P}_\pi[k]$ : Soundness reduction of Protocol  $\mathcal{P}_\pi$** 

1. If  $f(x, y) = 1$  then the prover sends the  $k$  quantum registers  $R_{j,i}$  ( $i = 1, \dots, k$ ), each of which has a state  $|h_x\rangle$  as certificate, to each of the intermediate nodes  $v_j$ ,  $j \in \{1, \dots, r-1\}$ .
2. The left-end node  $v_0$  prepares the  $k$  quantum registers  $R_{0,i}$ , each of which has  $|h_x\rangle$ .
3. For every  $j = 0, \dots, r-1$ , the node  $v_j$  chooses a  $k$ -bit string  $b_{j,1} \dots b_{j,k}$  uniformly at random, and sends  $R_{j,i}$ , together with the index  $i$ , to its right neighbor  $v_{j+1}$  whenever  $b_{j,i} = 0$ .
4. For every  $j = 1, \dots, r-1$  and for every  $i = 1, \dots, k$ , if the node  $v_j$  receives an index  $i$ , and if  $b_{j,i} = 1$ , then  $v_j$  performs the SWAP test on  $(R_{j-1,i}, R_{j,i})$ . Node  $v_j$  rejects whenever at least one of the performed SWAP tests rejects, and it accepts otherwise.
5. If the right-end node  $v_r$  receives an index  $i \in \{1, \dots, k\}$ , then it performs the POVM measurement  $\{M_{y,1}, M_{y,0}\}$  corresponding to  $\pi$  applied to the state in  $R_{r-1,i}$ . Node  $v_r$  rejects if at least one of the performed POVM measurements rejects, and it accepts otherwise.

Protocol  $\mathcal{P}_\pi[k]$  has completeness  $(1 - \varepsilon/2)^k$ , that is, the completeness has not reduced much whenever  $\varepsilon$  is small. By a similar analysis of standard error reduction techniques for quantum Merlin-Arthur games as the analysis in [3, 29], one can show that  $\mathcal{P}_\pi[k]$  has soundness  $(1 - 1/42r^2)^k$ . By choosing  $k = 84r^2$ , the resulting protocol  $\mathcal{P}_\pi[k]$  has completeness  $1 - 42r^2\varepsilon$  and soundness error  $(1/e)^2 < 1/3$ , while the size of the certificates is  $O(r^2q)$  qubits, and the length of the message exchanged between neighbors is  $O(r^2(q + \log r))$  (where the additional term  $\log r$  comes from the index to the right neighbor in Step 3 of  $\mathcal{P}_\pi[k]$ ).

Theorem 1 can now be easily derived from the above analysis. For the first part of the theorem, where  $f$  is having a one-sided-error one-way protocol  $\pi$ , simply use the protocol  $\mathcal{P}_\pi$  from Section 4.1 with  $\varepsilon = 0$ . The result then follows from the analysis of Section 4.2 and from the above discussion about soundness reduction.

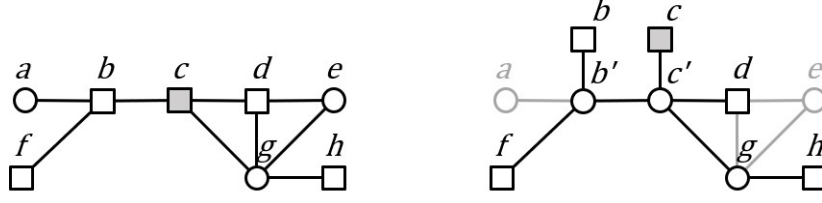
For the case of second part of the theorem, where  $f$  is having a two-sided-error one-way protocol, we repeat the protocol  $\pi$  for  $O(\log(n + r))$  times and using *majority voting* to get a protocol that correctly computes the value of the function with probability at least  $1 - 1/42n^c r^2$ . The protocol  $\pi$  of Section 4.1 can thus be chosen with  $\varepsilon = 1/42n^c r^2$ , with message size  $O(q \log(n + r))$ . The result then follows similarly. ◀

## 5 Certifying Equality in General Graphs

We now extend our protocol for checking equality between  $n$ -bit strings  $x_1, \dots, x_t$  stored at  $t \geq 2$  distinct nodes  $u_1, \dots, u_t$  of a connected simple graph  $G$ . We first show how to reduce the problem to trees of a specific structure, and then present a protocol for trees.

### 5.1 Reduction to Trees

Let  $G = (V, E)$  be a connected simple graph, and let  $u_1, \dots, u_t$  be  $t \geq 2$  distinct nodes of  $G$ . Assume, without loss of generality, that  $u_1$  is the most central node among them, i.e., it satisfies  $\max_{i=1, \dots, t} \text{dist}_G(u_1, u_i) = \min_{j=1, \dots, t} \max_{i=1, \dots, t} \text{dist}_G(u_j, u_i)$ . Let  $r = \max_{i=1, \dots, t} \text{dist}_G(u_1, u_i)$  be the *radius* of the  $t$  terminals  $u_1, \dots, u_t$ . We construct a tree  $T$  rooted at  $u_1$ , that has all terminals as leaves, maximum degree  $t$  and depth at most  $r + 1$  (see Figure 1). To this end, start with a BFS tree  $T'$  in  $G$ , rooted at  $u_1$ . Truncate the tree at each terminal  $u_i$  that does not have any terminal as successors, thus limiting



■ **Figure 1** The construction of a spanning tree: a graph  $G$  (left) and its corresponding spanning tree  $T$  (right). Terminals are marked by squares, and  $c$  is the root. Node  $b$  (resp.  $c$ ), which was a terminal, is replaced by a non-terminal node  $b'$  (resp.  $c'$ ), while the other terminals are leaves in the tree so they remain unmodified.

the depth to  $r$  and the degree to  $t$ . For every terminal  $u_i$  that is not a leaf, including  $u_1$ , replace  $u_i$  with a node  $u'_i$ , and connect  $u_i$  to  $u'_i$  as a leaf, where the input  $x_i$  stays at  $u_i$  – this guarantees that all inputs are now on leaves, the same degree bound holds, and the depth is increased by at most 1.

While  $T$  is not a sub-tree of  $G$ , we can easily emulate an algorithm or a labeling scheme designed for  $T$ , in  $G$  (specifically, in  $T'$ ). To this end, every internal terminal  $u_i$  in  $T'$  simulates the behavior of  $u_i$  itself, and also of  $u'_i$ . The following lemma is using classical assumptions of network computing (see, e.g., [42]) and can be proved using standard techniques (see, e.g., [35]). We refer to the tree  $T$  in the construction described above.

► **Lemma 7.** *For any graph  $G = (V, E)$  with nodes IDs taken in a range polynomial in  $|V|$ , there is a deterministic distributed Merlin-Arthur protocol for the tree  $T$  using certificates on  $O(\log |V|)$  bits.*

The term *deterministic* in the above lemma means that the verification process is deterministic, which implies perfect completeness and perfect soundness (i.e., soundness error 0). Roughly speaking, in this protocol each non-tree node will have a (non-quantum) label indicating its distance from the tree, and each tree node will have as label its depth in the tree, the ID of its parent, and the ID of the root.

## 5.2 Certifying Equality in Trees

Based on our tree construction from a graph and Lemma 7, we can restrict our attention to the case in which the  $t$  terminals  $u_1, \dots, u_t$ , who hold the  $n$ -bit strings  $x_1, \dots, x_t$ , belong to a tree  $T$  rooted at  $u_1$ , of depth equal to  $r + 1$ , where  $r$  is the radius of the terminals, with maximum degree  $t$ , and with leaves  $u_2, \dots, u_t$ . Moreover, we assume that the root  $u_1$  itself is of degree 1 due to our tree construction. We present a distributed quantum Merlin-Arthur protocol for the equality function  $\text{EQ}_n^t$  in this setting, and hence prove our main result.

► **Theorem 8.** *There is a distributed quantum Merlin-Arthur protocol on  $T$  for  $\text{EQ}_n^t$  between  $t$  terminals of radius  $r$ , with perfect completeness, soundness  $1/3$ , certificate size  $O(t r^2 \log n)$  qubits, and message length  $O(t r^2 \log(n + r))$  qubits.*

**Proof.** Let  $\pi$  be a one-way communication protocol for  $\text{EQ}_n$  transmitting  $m = O(\log n)$  qubits such that, for every input pair  $(x, y)$ , if  $x = y$  then  $\pi$  outputs 1 with probability 1 and if  $x \neq y$  then  $\pi$  outputs 0 with probability at least  $2/3$  (such a protocol exists, as mentioned in Section 2). For input  $(x, y)$ , let  $|h_x\rangle$  be the quantum message from Alice to Bob in  $\pi$ , and let

$\{M_{y,1}, M_{y,0}\}$  be the POVM measurement done by Bob on  $|h_x\rangle$  in  $\pi$ , where  $M_{y,1}$  corresponds to the measurement result 1 (accept), and  $M_{y,0}$  corresponds to the measurement result 0 (reject), respectively. Our quantum Merlin-Arthur protocol is as follows.

**Protocol  $\mathcal{P}(\text{EQ}_n^t)$  for equality in trees**

1. If  $\text{EQ}_n^t(x_1, \dots, x_t) = 1$  then the prover sends an  $m$ -qubit state equal to  $|h_{x_1}\rangle$  to each of the nodes  $v$  that have no input.
2. For every  $i \in \{1, \dots, t\}$ , node  $u_i$  prepares the  $m$ -qubit state  $|h_{x_i}\rangle$ .
3. Every non-root node  $v$  of the tree chooses a bit  $b_v$  uniformly at random. If  $b_v = 0$ , then  $v$  sends its  $m$ -qubit state to its parent in  $T$ .
4. For every non-terminal node  $v$ , if  $v$  receives a state from the children, and if  $b_v = 1$ , then  $v$  performs the SWAP test on the  $2m$ -qubit state that consists of the  $m$ -qubit state received from the prover and an  $m$ -qubit state received from the children, which is chosen uniformly at random if he/she receives multiple  $m$ -qubit states from the children, and accepts or rejects accordingly. Otherwise,  $v$  accepts.
5. If the root node  $u_1$  receives a state from its children, then  $u_1$  performs the POVM measurement  $\{M_{x_1,1}, M_{x_1,0}\}$  on an  $m$ -qubit state received from the children, which is chosen uniformly at random if he/she receives multiple  $m$ -qubit states from the children, and accepts or rejects accordingly. Otherwise,  $u_1$  accepts.

The perfect completeness trivially holds since every local test yields acceptance with certainty. For the soundness, if  $\text{EQ}_n^t(x_1, \dots, x_t) = 0$  then there is a leaf  $u_i$ ,  $i > 1$ , with  $x_i \neq x_1$ . Then, we can perform almost the same analysis as in Section 4, but for the path connecting  $u_1$  and  $u_i$  in  $T$ . The only difference is the probability that each local test occurs; it is at least  $1/4$  in the analysis of  $\mathcal{P}_\pi$  done in Section 4, while it is at least  $(1/4) \cdot (1/t)$  in the protocol  $\mathcal{P}(\text{EQ}_n^t)$  we are now considering, as every non-terminal node  $v_j$  or  $u_1$  on the path chooses the  $m$ -qubit state from the child on the path uniformly at random from the multiple  $m$ -qubit states (possibly) sent from all the children. Hence,  $\mathcal{P}(\text{EQ}_n^t)$  has soundness error  $1 - O(1/tr^2)$ . The proof is completed by performing  $O(tr^2)$  parallel repetitions of  $\mathcal{P}(\text{EQ}_n^t)$  for error reduction, similarly to the  $O(r^2)$  parallel repetitions of  $\mathcal{P}_\pi$  in Section 4. ◀

► **Remark.** Using Lemma 7, we get that, up to adding  $O(\log |V|)$  classical bits in the certificates of the nodes, Theorem 8 can be extended to the case where the terminals are in a connected graph  $G = (V, E)$ .

## 6 Classical Lower Bounds

In this section, we show that non-quantum distributed Merlin-Arthur (dMA) protocols for distant EQ require certificates of linear size. In fact, we establish a more general lower bound which applies to all functions  $f$  with large fooling set, even using shared randomness. In addition, the bound holds for settings which allow the graph nodes to have multiple communication rounds among them, after receiving the certificates and before deciding if they finally accept (see, e.g., [19, 41]).

For the lower bound, it is sufficient to consider the path  $v_0, \dots, v_r$  in which  $v_0$  and  $v_r$  are provided with inputs  $x$  and  $y$ , respectively.

► **Theorem 9.** *Let  $r \geq 2\mu + 1$ , and let  $f(x, y)$  be any Boolean function with a 1-fooling set of size at least  $k$ . Let  $\mathcal{P}$  be a classical Merlin-Arthur protocol for  $f$  in a path of  $r$  edges, with  $\mu$  rounds of communication among the nodes, shared randomness, certificates of size  $\lfloor \frac{1}{2\mu} \log(k-1) \rfloor$  bits, and completeness  $1 - p$ . Then  $\mathcal{P}$  has soundness error at least  $1 - 2p$ .*

**Proof.** Consider the path  $v_0, \dots, v_r$  with a fixed identifier assignment, and inputs  $x$  and  $y$  given to  $v_0$  and  $v_r$ , respectively. Since  $f$  has large 1-fooling set but small certificates, there exist two distinct pairs of “fooling” inputs that have the same certificate assignments on the  $2\mu$  node  $v_1, \dots, v_{2\mu}$ . That is, we can fix two input pairs  $(x, y)$  and  $(x', y')$ , with  $f(x, y) = f(x', y') = 1$ , and, w.l.o.g.,  $f(x, y') = 0$ , with corresponding certificate assignments  $w$  and  $w'$ , such that

$$w(v_i) = w'(v_i) \text{ for every } i \in \{1, \dots, 2\mu\},$$

where  $w(v_i)$  and  $w'(v_i)$  are the certificate assigned to the node  $v_i$  in the assignments  $w$  and  $w'$ , respectively.

We interpret the outputs as Boolean values, where accept = 1 and reject = 0, and denote by  $\text{out}_i(x, y, w)$  the output of  $v_i$  when the inputs are  $x$  and  $y$  and the certificate assignment is  $w$ . Since  $\mathcal{P}$  has completeness  $1 - p$ , we have

$$\Pr_s \left[ \bigwedge_{i \leq \mu} \text{out}_i(x, y, w) = 1 \wedge \bigwedge_{i \geq \mu+1} \text{out}_i(x, y, w) = 1 \right] \geq 1 - p,$$

and the same holds for  $(x', y', w')$ . Hence,

$$\Pr_s \left[ \bigwedge_{i \leq \mu} \text{out}_i(x, y, w) = 1 \right] \geq 1 - p, \text{ and } \Pr_s \left[ \bigwedge_{i \geq \mu+1} \text{out}_i(x', y', w') = 1 \right] \geq 1 - p.$$

The output  $\text{out}_i$  of every node  $v_i$  is a function of its own identifier and certificate, the certificates of the nodes in its distance- $\mu$  neighborhood, and the public random string  $s$ . In addition, the outputs of  $v_0, v_1, \dots, v_\mu$  may also depend on the input  $x$  to  $v_0$ , and the outputs of  $v_{r-\mu}, \dots, v_r$  may also depend on the input  $y$  to  $v_r$ . Formally speaking, the outputs can also depend on the identifiers of the neighbors, but these are fixed given the node’s identifier, so we ignore them henceforth.

Let  $w''$  be the certificate assignment defined by

$$\begin{aligned} w''(v_0) &= w(v_0); \\ w''(v_i) &= w(v_i) = w'(v_i) \quad \text{for } i \in \{1, \dots, 2\mu\}; \\ w''(v_i) &= w'(v_i) \quad \text{for } i \in \{2\mu+1, \dots, r\}. \end{aligned}$$

Consider the input assignment  $(x, y')$  combined with the certificate assignment  $w''$ . The definition of  $w''$  implies that nodes  $v_0, \dots, v_{2\mu}$  receive the same certificates as in  $w$ , and thus nodes  $v_0, \dots, v_\mu$  cannot distinguish this from the input assignment  $(x, y)$  with certificates assignment  $w$ . On the other hand, nodes  $v_1, \dots, v_r$  receive the same certificates as in  $w'$ , so the nodes  $v_{\mu+1}, \dots, v_r$  cannot distinguish this from the input assignment  $(x', y')$  with certificates assignment  $w'$ .

A union bound finishes the proof:

$$\begin{aligned} & \Pr_s \left[ \bigwedge_{i \leq \mu} \text{out}_i(x, y', w'') = 1 \wedge \bigwedge_{i \geq \mu+1} \text{out}_i(x, y', w'') = 1 \right] \\ &= \Pr_s \left[ \bigwedge_{i \leq \mu} \text{out}_i(x, y, w) = 1 \wedge \bigwedge_{i \geq \mu+1} \text{out}_i(x', y', w') = 1 \right] \\ &\geq 1 - \Pr_s \left[ \neg \bigwedge_{i \leq \mu} \text{out}_i(x, y, w) = 1 \right] - \Pr_s \left[ \neg \bigwedge_{i \geq \mu+1} \text{out}_i(x', y', w') = 1 \right] \\ &\geq 1 - 2p. \end{aligned}$$

That is, we found a certificate assignment  $w''$  for the input  $(x, y')$  which makes all node accept with probability at least  $1 - 2p$ , even though  $f(x, y') = 0$ . Hence, the soundness error is at least  $1 - 2p$ , as claimed.  $\blacktriangleleft$

Since  $\text{EQ}_n$  has a 1-fooling set of size  $2^n$ , the corollary below follows directly from Theorem 9. The case  $r = 3$  and  $\mu = 1$  gives a linear lower bound for dMA protocols on a 4-node path.

► **Corollary 10.** *For every  $r \geq 2\mu + 1$ , every distributed (classical) Merlin-Arthur protocol for  $\text{EQ}_n^2$  with  $\mu$  rounds of communication among the nodes in the path of  $r$  edges with certificates of size at most  $\lfloor (n-1)/2\mu \rfloor$  bits, and completeness  $1-p$  has soundness error at least  $1-2p$ .*

The requirements for a good protocol is to have a high completeness (i.e., small value for  $p$ , ideally  $p = 0$ ) and a reasonably small soundness error. Corollary 10 precisely shows that for the equality function such protocols cannot exist in the classical setting unless the certificate size is linear in  $n$ .

► **Remark.** The completeness-soundness gap of Theorem 9 is optimal in general, in the sense that it cannot be improved for  $\text{EQ}_1^2$ , i.e., distant equality between two input bits. Consider the following protocol  $\mathcal{P}$  for  $\text{EQ}_1^2$ , on input  $(x, y) \in \{0, 1\} \times \{0, 1\}$ . It uses a shared random variable  $X \in \{-1, 0, 1\}$  with  $\Pr[X = 0] = \Pr[X = 1] = p$ , and  $\Pr[X = -1] = 1 - 2p$ . In the case  $X = -1$ , all the nodes accept. In the case  $X \in \{0, 1\}$ ,  $v_0$  accepts whenever  $X = x$ ,  $v_r$  accepts whenever  $X = y$ , and all the other nodes accept. If  $x = y$ , the probability that all the nodes accept is  $1 - 2p + (1/2) \cdot (2p) = 1 - p$ . If  $x \neq y$ , then either  $v_0$  or  $v_r$  systematically rejects for  $X \neq -1$ , and thus the probability that all the nodes accept is  $1 - 2p$ .

## 7 Conclusion

In this paper, we extended the notion of randomized proof-labeling scheme to the quantum setting. We showed the efficiency of distributed quantum certification mechanisms by designing a distributed quantum Merlin-Arthur protocol for  $\text{EQ}_n^t$  between  $t$  parties spread out in a graph, using certificates and messages whose size depend logarithmically on  $n$ , the size of the data. This is in contrast to classical distributed Merlin-Arthur protocols, which require certificates of size linear in  $n$ , even when messages of unbounded size can be used. Our result was obtained by using an interesting property of the SWAP test: it can be applied for checking proximity properties between reduced states.

Which other Boolean predicates on labeled graphs, beyond equality, could take benefit from quantum resources for the design of compact distributed certification schemes is an intriguing question. Theorem 1 gives a partial answer on the path. A complete answer to this question would significantly help improving our comprehension of the power of quantum computing in the distributed setting.

## References

- 1 Scott Aaronson, Salman Beigi, Andrew Drucker, Bill Fefferman, and Peter W. Shor. The power of unentanglement. *Theory of Computing*, 5:1:1–1:42, 2009. doi:10.4086/toc.2009.v005a001.
- 2 Yehuda Afek, Shay Kutten, and Moti Yung. The local detection paradigm and its application to self-stabilization. *Theoretical Computer Science*, 186(1-2):199–229, 1997. doi:10.1016/S0304-3975(96)00286-1.
- 3 Dorit Aharonov and Tomer Naveh. Quantum NP – a survey. arXiv:quant-ph/0210077v1, 2002. arXiv:quant-ph/0210077.
- 4 Noga Alon, Klim Efremenko, and Benny Sudakov. Testing equality in communication graphs. *IEEE Transactions on Information Theory*, 63(11):7569–7574, 2017. doi:10.1109/TIT.2017.2744608.
- 5 Heger Arfaoui and Pierre Fraigniaud. What can be computed without communications? *SIGACT News*, 45(3):82–104, 2014. doi:10.1145/2670418.2670440.

- 6 Baruch Awerbuch, Boaz Patt-Shamir, and George Varghese. Self-stabilization by local checking and correction (extended abstract). In *32nd Symposium on Foundations of Computer Science (FOCS)*, pages 268–277, 1991. doi:10.1109/SFCS.1991.185378.
- 7 Alkida Balliu, Gianlorenzo D’Angelo, Pierre Fraigniaud, and Dennis Olivetti. What can be verified locally? *Journal of Computer System and Sciences*, 97:106–120, 2018. doi:10.1016/j.jcss.2018.05.004.
- 8 Michael Ben-Or and Avinatan Hassidim. Fast quantum byzantine agreement. In *37th Annual ACM Symposium on Theory of Computing (STOC)*, pages 481–485, 2005. doi:10.1145/1060590.1060662.
- 9 Hugue Blier and Alain Tapp. A quantum characterization of NP. *Computational Complexity*, 21(3):499–510, 2012. doi:10.1007/s00037-011-0016-2.
- 10 Ralph Bottesch, Dmitry Gavinsky, and Hartmut Klauck. Equality, revisited. In *40th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 9235 of *LNCS*, pages 127–138. Springer, 2015. doi:10.1007/978-3-662-48054-0\_11.
- 11 Anne Broadbent and Alain Tapp. Can quantum mechanics help distributed computing? *SIGACT News*, 39(3):67–76, 2008. doi:10.1145/1412700.1412717.
- 12 Harry Buhrman, Richard Cleve, John Watrous, and Ronald de Wolf. Quantum fingerprinting. *Physical Review Letters*, 87(16):167902:1–167902:4, 2001. doi:10.1103/PhysRevLett.87.167902.
- 13 Arkadev Chattopadhyay, Jaikumar Radhakrishnan, and Atri Rudra. Topology matters in communication. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS*, pages 631–640, 2014. doi:10.1109/FOCS.2014.73.
- 14 Arkadev Chattopadhyay and Atri Rudra. The range of topological effects on communication. In *Automata, Languages, and Programming - 42nd International Colloquium, ICALP*, pages 540–551, 2015. doi:10.1007/978-3-662-47666-6\_43.
- 15 Pierluigi Crescenzi, Pierre Fraigniaud, and Ami Paz. Trade-offs in distributed interactive proofs. In *33rd International Symposium on Distributed Computing (DISC)*, pages 13:1–13:17, 2019. doi:10.4230/LIPIcs.DISC.2019.13.
- 16 Vasil S. Denchev and Gopal Pandurangan. Distributed quantum computing: a new frontier in distributed systems or science fiction? *SIGACT News*, 39(3):77–95, 2008. doi:10.1145/1412700.1412718.
- 17 Michael Elkin, Hartmut Klauck, Danupon Nanongkai, and Gopal Pandurangan. Can quantum communication speed up distributed computation? In *33rd ACM Symposium on Principles of Distributed Computing (PODC)*, pages 166–175, 2014. doi:10.1145/2611462.2611488.
- 18 Laurent Feuilloley, Pierre Fraigniaud, and Juho Hirvonen. A hierarchy of local decision. In *43rd International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 118:1–118:15, 2016. doi:10.4230/LIPIcs.ICALP.2016.118.
- 19 Laurent Feuilloley, Pierre Fraigniaud, Juho Hirvonen, Ami Paz, and Mor Perry. Redundancy in distributed proofs. In *32nd International Symposium on Distributed Computing, DISC*, pages 24:1–24:18, 2018. doi:10.4230/LIPIcs.DISC.2018.24.
- 20 Pierre Fraigniaud, Amos Korman, and David Peleg. Towards a complexity theory for local distributed computing. *Journal of the ACM*, 60(5):35:1–35:26, 2013. doi:10.1145/2499228.
- 21 Pierre Fraigniaud, Pedro Montealegre, Rotem Oshman, Ivan Rapaport, and Ioan Todinca. On distributed Merlin-Arthur decision protocols. In *26th International Colloquium on Structural Information and Communication Complexity (SIROCCO)*, volume 11639 of *LNCS*, pages 230–245. Springer, 2019. doi:10.1007/978-3-030-24922-9\_16.
- 22 Pierre Fraigniaud, Boaz Patt-Shamir, and Mor Perry. Randomized proof-labeling schemes. *Distributed Computing*, 32(3):217–234, 2019. doi:10.1007/s00446-018-0340-8.
- 23 Cyril Gavoille, Adrian Kosowski, and Marcin Markiewicz. What can be observed locally? In *23rd International Symposium on Distributed Computing (DISC)*, volume 5805 of *LNCS*, pages 243–257. Springer, 2009. doi:10.1007/978-3-642-04355-0\_26.



- 24 Mika Göös and Jukka Suomela. Locally checkable proofs in distributed computing. *Theory of Computing*, 12:19:1–19:33, 2016. doi:10.4086/toc.2016.v012a019.
- 25 Aram W. Harrow and Ashley Montanaro. Testing product states, quantum Merlin-Arthur games and tensor optimization. *Journal of the ACM*, 60(1):3:1–3:43, 2013. doi:10.1145/2432622.2432625.
- 26 Gene Itkis and Leonid A. Levin. Fast and lean self-stabilizing asynchronous protocols. In *35th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 226–239, 1994. doi:10.1109/SFCS.1994.365691.
- 27 Taisuke Izumi and François Le Gall. Quantum distributed algorithm for the All-Pairs Shortest Path problem in the CONGEST-CLIQUE model. In *38th ACM Symposium on Principles of Distributed Computing (PODC)*, pages 84–93, 2019. doi:10.1145/3293611.3331628.
- 28 Taisuke Izumi, François Le Gall, and Frédéric Magniez. Quantum distributed algorithm for triangle finding in the CONGEST model. In *37th International Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 23:1–23:13, 2020. doi:10.4230/LIPIcs.STACS.2020.23.
- 29 Alexei Yu. Kitaev, Alexander H. Shen, and Mikhail N. Vyalyi. *Classical and Quantum Computation*. Graduate Studies in Mathematics 47. American Mathematical Society, 2002.
- 30 Hartmut Klauck. On Arthur Merlin games in communication complexity. In *26th Annual IEEE Conference on Computational Complexity (CCC)*, pages 189–199, 2011. doi:10.1109/CCC.2011.33.
- 31 Hartmut Klauck and Supartha Podder. Two results about quantum messages. In *39th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 8635 of *LNCS*, pages 445–456. Springer, 2014. doi:10.1007/978-3-662-44465-8\_38.
- 32 Hirotada Kobayashi, François Le Gall, and Harumichi Nishimura. Stronger methods of making quantum interactive proofs perfectly complete. *SIAM Journal on Computing*, 44(2):243–289, 2015. doi:10.1137/140971944.
- 33 Hirotada Kobayashi, Keiji Matsumoto, and Tomoyuki Yamakami. Quantum Merlin-Arthur proof systems: Are multiple Merlins more helpful to Arthur? *Chicago Journal of Theoretical Computer Science*, 2009:3:1–3:19, 2009. doi:10.4086/cjtcs.2009.003.
- 34 Gillat Kol, Rotem Oshman, and Raghuvansh R. Saxena. Interactive distributed proofs. In *37th ACM Symposium on Principles of Distributed Computing (PODC)*, pages 255–264, 2018. doi:10.1145/3212734.3212771.
- 35 Amos Korman, Shay Kutten, and David Peleg. Proof labeling schemes. *Distributed Computing*, 22(4):215–233, 2010. doi:10.1007/s00446-010-0095-3.
- 36 Eyal Kushilevitz and Noam Nisan. *Communication Complexity*. Cambridge University Press, 1997.
- 37 François Le Gall and Frédéric Magniez. Sublinear-time quantum computation of the diameter in CONGEST networks. In *37th ACM Symposium on Principles of Distributed Computing (PODC)*, pages 337–346, 2018. doi:10.1145/3212734.3212744.
- 38 François Le Gall, Harumichi Nishimura, and Ansis Rosmanis. Quantum advantage for the LOCAL model in distributed computing. In *36th International Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 49:1–49:14, 2019. doi:10.4230/LIPIcs.STACS.2019.49.
- 39 Moni Naor, Merav Parter, and Eylon Yogev. The power of distributed verifiers in interactive proofs. In *31st ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1096–1115, 2020. doi:10.1137/1.9781611975994.67.
- 40 Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.
- 41 Rafail Ostrovsky, Mor Perry, and Will Rosenbaum. Space-time tradeoffs for distributed verification. In *Structural Information and Communication Complexity - 24th International Colloquium, SIROCCO*, pages 53–70, 2017. doi:10.1007/978-3-319-72050-0\_4.
- 42 David Peleg. *Distributed Computing: A Locality-Sensitive Approach*. SIAM, 2000.



- 43 Ran Raz and Amir Shpilka. On the power of quantum proofs. In *19th IEEE Conference on Computational Complexity (CCC)*, pages 260–274, 2004. doi:10.1109/CCC.2004.1313849.
- 44 Bill Rosgen. Distinguishing short quantum computations. In *25th International Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 597–608, 2008. doi:10.4230/LIPIcs.STACS.2008.1322.
- 45 Seiichiro Tani, Hirotada Kobayashi, and Keiji Matsumoto. Exact quantum algorithms for the leader election problem. *ACM Transactions on Computation Theory*, 4(1):1:1–1:24, 2012. doi:10.1145/2141938.2141939.
- 46 Andrew Chi-Chih Yao. Quantum circuit complexity. In *34th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 352–361, 1993. doi:10.1109/SFCS.1993.366852.
- 47 Andrew Chi-Chih Yao. On the power of quantum fingerprinting. In *35th ACM Symposium on Theory of Computing (STOC)*, pages 77–81, 2003. doi:10.1145/780542.780554.

# On Basing Auxiliary-Input Cryptography on NP-Hardness via Nonadaptive Black-Box Reductions

Mikito Nanashima

Tokyo Institute of Technology, Japan

<https://nanashima.github.io>

[nanashima.m.aa@is.c.titech.ac.jp](mailto:nanashima.m.aa@is.c.titech.ac.jp)

---

## Abstract

Constructing one-way functions based on NP-hardness is a central challenge in theoretical computer science. Unfortunately, Akavia et al. [2] presented strong evidence that a nonadaptive black-box (BB) reduction is insufficient to solve this challenge. However, should we give up such a central proof technique even for an intermediate step?

In this paper, we turn our eyes from standard cryptographic primitives to weaker cryptographic primitives allowed to take auxiliary-input and continue to explore the capability of nonadaptive BB reductions to base auxiliary-input primitives on NP-hardness. Specifically, we prove the followings:

- if we base an auxiliary-input pseudorandom generator (AIPRG) on NP-hardness via a nonadaptive BB reduction, then the polynomial hierarchy collapses;
- if we base an auxiliary-input one-way function (AIOWF) or auxiliary-input hitting set generator (AIHSG) on NP-hardness via a nonadaptive BB reduction, then an (i.o.-)one-way function also exists based on NP-hardness (via an adaptive BB reduction).

These theorems extend our knowledge on nonadaptive BB reductions out of the current worst-to-average framework. The first result provides new evidence that nonadaptive BB reductions are insufficient to base AIPRG on NP-hardness. The second result also yields a weaker but still surprising consequence of nonadaptive BB reductions, i.e., a one-way function based on NP-hardness. In fact, the second result is interpreted in the following two opposite ways. Pessimistically, it shows that basing AIOWF or AIHSG on NP-hardness via nonadaptive BB reductions is harder than constructing a one-way function based on NP-hardness, which can be regarded as a negative result. Note that AIHSG is a weak primitive implied even by the hardness of learning; thus, this pessimistic view provides conceptually stronger limitations than the currently known limitations on nonadaptive BB reductions. Optimistically, it offers a new hope: breakthrough construction of auxiliary-input primitives might also provide construction standard cryptographic primitives. This optimistic view enhances the significance of further investigation on constructing auxiliary-input or other intermediate cryptographic primitives instead of standard cryptographic primitives.

**2012 ACM Subject Classification** Theory of computation → Computational complexity and cryptography

**Keywords and phrases** Auxiliary-input cryptographic primitives, nonadaptive black-box reductions

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.29

**Related Version** A full version of the paper is available at <https://eccc.weizmann.ac.il/report/2020/095/>.

**Funding** Mikito Nanashima: JST, ACT-X Grant Number JPMJAX190M, Japan.

**Acknowledgements** The author thanks Toshiya Itoh and the anonymous reviewers for many helpful comments.



© Mikito Nanashima;

licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 29; pp. 29:1–29:15



Leibniz International Proceedings in Informatics

LIPIcs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

## 1 Introduction

How can we translate computational hardness into cryptography? This is a central question in theoretical computer science. Specifically, one of the most significant and long-standing challenges on this question is constructing fundamental cryptographic primitives such as a one-way function based on NP-hardness. At the moment, several breakthroughs seem to be required for solving this challenge, as surveyed by Impagliazzo [20].

A central ingredient for solving the above challenge is a *reduction*; in other words, the way to translate recognizing a language into breaking a cryptographic primitive. A reduction is a powerful proof technique even if it is restricted to a quite simple form, and in fact, a nonadaptive black-box (BB) reduction is sufficient to show many brilliant results and has played a crucial role in theoretical computer science. Therefore, it is a natural attempt to apply such a familiar proof technique even for constructing secure cryptographic primitives.

However, Akavia et al. [10] presented strong evidence that such a simple reduction is insufficient for cryptography based on NP-hardness. Generally, breaking cryptographic primitives is formulated as an NP problem on an efficiently samplable distribution that is fixed in advance. They showed that there is no nonadaptive BB reduction from an NP-hard problem to such a distributional NP problem unless the polynomial hierarchy collapses. As a corollary, their work excluded the attempt to apply nonadaptive BB reductions for cryptography based on NP-hardness under the reasonable assumption that the polynomial hierarchy does not collapse. Further, subsequent work provided stronger consequences in more specific cases of several cryptographic primitives [2, 12, 5, 13, 9, 8, 27, 17].

Then should we also give up all nonadaptive BB strategies even for an intermediate step towards cryptography? This question originally motivated our work. In this spirit, we focus on the capability of nonadaptive BB reduction for a weaker cryptographic notion, i.e., an *auxiliary-input* cryptographic primitive introduced first by Ostrovsky and Wigderson [30]. Informally speaking, an auxiliary-input cryptographic primitive is defined as a family of primitives indexed by the auxiliary-input and has a relaxed security requirement: at least one primitive in the family must be secure depending on each adversary (instead of one specific primitive secure against all adversaries). In other words, adversaries for auxiliary-input primitives must break all primitives in the worst-case sense on auxiliary-input. This task is not directly formulated as a distributional NP-problem because the distribution is not uniquely determined in advance due to auxiliary-input. Thus, the previous work on distributional NP problem cannot be directly applied to auxiliary-input cryptography.

Herein, we present the current status of nonadaptive BB reductions to auxiliary-input cryptography. Applebaum et al. [5] observed that nonadaptive fixed-auxiliary-input BB reductions are insufficient even for auxiliary-input cryptography unless the polynomial hierarchy collapses. Their reduction is a restricted case of nonadaptive BB reduction where only one auxiliary-input is accessible. However, this restricted access to auxiliary-input seems too strict and implicitly yields a reduction from an NP-hard language to some fixed cryptographic primitive (depending on the instance). In fact, this result was shown in almost the same way to the previous result for standard cryptographic primitives in [2]. The same work and later Xiao [34] observed that generalizing their result to nonadaptive BB reductions seems hard by giving the explicit technical issue. To the best of our knowledge, we had no negative result on general nonadaptive BB reductions to base auxiliary-input cryptography on NP-hardness before this work. For more detailed reason why the previous work such as [10, 2] is not applicable for auxiliary-input primitives, refer to Section 4.

The recent progress on the minimum circuit size problem revealed that an auxiliary-input one-way function indeed implies a hard-on-average distributional NP problem [3, 16]. However, such an implication requires adaptive techniques at present (e.g., [15]). Thus, the property of nonadaptive black-box is lost in translating reductions for the auxiliary-input primitive into reductions for the distributional NP problem.

In this paper, based on the above status, we continue to investigate the capability of nonadaptive BB reductions for auxiliary-input cryptographic primitives based on NP-hardness. The importance of our work is to extend our current knowledge on such a central proof technique out of the previous worst-to-average framework in [10] and to identify the inherent difficulty on constructing cryptographic primitives on NP-hardness more finely.

## 1.1 Our Contribution

Our main contribution is to provide new knowledge about nonadaptive BB reductions from an NP-hard problem to an auxiliary-input cryptographic primitive. In particular, we handle the auxiliary-input analogs of the following three fundamental primitives: a one-way function, a pseudorandom generator, and a hitting set generator. A definition of each primitive will be presented in Section 2 with a formal description of our main results. First, we informally state the main theorem as follows.

► **Theorem (informal).** *If there is a nonadaptive BB reduction from an NP-hard language  $L$  to breaking an auxiliary-input cryptographic primitive  $P$ , then the following statements hold according to the type of  $P$ :*

- *if  $P$  is an auxiliary-input pseudorandom generator, then the polynomial hierarchy collapses;*
- *if  $P$  is an auxiliary-input one-way function or an auxiliary-input hitting set generator, then there is also an adaptive reduction from  $L$  to inverting some (i.o.-)one-way function.*

The first result provides reasonable evidence that auxiliary-input pseudorandom generators (AIPRG) cannot be based on NP-hardness via nonadaptive BB reductions as standard cryptography. The second result shows that a nonadaptive BB reduction for basing the other auxiliary-input primitives yields another strong consequence: an “infinitely often” analog of one-way function based on NP-hardness. Note that an auxiliary-input hitting set generator (AIHSG) is much weaker primitive than standard cryptographic primitives: for example, the existence is even weaker than the hardness of PAC learning [28]. What is surprising is that even a nonadaptive BB reduction to such a weak primitive yields a solution close to the long-standing challenge, i.e., characterization of one-way functions based on NP-hardness.

The second result is not sufficient to exclude nonadaptive BB reductions which base auxiliary-input primitives on NP-hardness, and it has two opposite interpretations. However, let us stress that both interpretations are quite nontrivial and yield new knowledge about nonadaptive BB reductions. One interpretation is a pessimistic (or realistic) one. As mentioned in the introduction, no one has not come up with the construction of a one-way function based on NP-hardness for several decades despite its importance. Thus, this result is still strong evidence of difficulty finding such a simple reduction. The other interpretation is an optimistic one as a new approach to constructing a one-way function. We will further discuss this optimistic perspective and its novelty in Section 3.

A reader who is familiar with cryptography may wonder why the consequences are different between an auxiliary-input one-way function (AIOWF) and AIPRG. In fact, AIPRG is constructed from any AIOWF by applying the known BB construction of a pseudorandom generator from a one-way function. However, if such construction requires an adaptive security proof, then the property of nonadaptive is lost in translating reductions for AIOWF

into reductions for AIPRG via the adaptive security reduction. To the best of our knowledge, all currently known constructions of pseudorandom generators (e.g., [15, 19, 14]) use adaptive techniques in the security proof; for instance, construction of false entropy generators and the uniform hardcore lemma [18]. This technical issue prevents us from applying the first result for AIPRG to AIOWF. For a similar reason, our second result on AIOWF is incomparable with the previous work [5] on hardness of learning because we need to construct AIPRG first to show the hardness of learning from the existence of AIOWF.

## 2 Formal Descriptions

In this section, we present formal descriptions of auxiliary-input primitives and our results. Let us introduce a few notations. For any  $n \in \mathbb{N}$ , let  $U_n$  denote a random variable selected according to a uniform distribution over  $\{0, 1\}^n$ . For any function  $f : \mathcal{X} \rightarrow \mathcal{Y}$  and subsets  $X \subseteq \mathcal{X}$ ,  $Y \subseteq \mathcal{Y}$ , let  $f(X) = \{f(x) : x \in X\}$  and  $f^{-1}(Y) = \{x \in X : f(x) \in Y\}$ . For a language  $L$ , let  $(L, U)$  denote a distributional problem of recognizing  $L(x)$  on an instance  $x$  selected uniformly at random. An auxiliary-input cryptographic primitive is defined as an auxiliary-input function with some additional security conditions.

► **Definition 1** (Auxiliary-input function). *A (polynomial-time computable) auxiliary-input function is a family  $f = \{f_z : \{0, 1\}^{n(|z|)} \rightarrow \{0, 1\}^{\ell(|z|)}\}_{z \in \{0, 1\}^*}$ , where  $n(|z|)$  and  $\ell(|z|)$  are polynomially-related<sup>1</sup> to  $|z|$ , which satisfies that there exists a polynomial-time evaluation algorithm  $F$  such that for any  $z \in \{0, 1\}^*$  and  $x \in \{0, 1\}^{n(|z|)}$ ,  $F(z, x)$  outputs  $f_z(x)$ .*

In this paper, we use the term “an auxiliary-input function (AIF)” to refer to polynomial-time computable one as in the above definition unless otherwise stated. For the sake of simplicity, we assume that  $n(\cdot)$  and  $\ell(\cdot)$  are increasing functions. Note that the length of auxiliary-input is possibly longer than the length of input and output, i.e.,  $|z| > n(|z|)$  and  $|z| > \ell(|z|)$ . We may write  $n(|z|)$  (resp.  $\ell(|z|)$ ) as  $n$  (resp.  $\ell$ ) when the dependence of  $|z|$  is obvious.

### 2.1 Auxiliary-Input Pseudorandom Generator

A pseudorandom generator is a primitive stretching a short random seed to a long binary string random-looking from all efficiently computable adversaries. The auxiliary-input analog is formally defined as follows:

► **Definition 2** (Auxiliary-input pseudorandom generator). *Let  $G = \{G_z : \{0, 1\}^n \rightarrow \{0, 1\}^{\ell(n)}\}_{z \in \{0, 1\}^*}$  be an auxiliary-input function. For a function  $\gamma : \mathbb{N} \rightarrow (0, 1)$ , we say that a randomized algorithm  $A$   $\gamma$ -distinguishes  $G$  if for all auxiliary-inputs  $z \in \{0, 1\}^*$ ,*

$$\left| \Pr_{A, U_n} [A(z, G_z(U_n)) = 1] - \Pr_{A, U_{\ell(n)}} [A(z, U_{\ell(n)}) = 1] \right| \geq \gamma(n).$$

*We say that  $G$  is an auxiliary-input pseudorandom generator (AIPRG) if  $\ell(n) > n$  and for all polynomials  $p$ , there exists no polynomial-time randomized algorithm  $(1/p)$ -distinguishing  $G$ .*

A BB reduction for AIPRG is defined as follows. It is easily verified that the following BB reduction from a language  $L$  to distinguishing an AIF  $G$  shows that  $G$  is an AIPRG if  $L \notin \text{BPP}$ .

<sup>1</sup> In the case of  $n(|z|)$ , it means that there exist  $c, c' \in \mathbb{N}$  such that  $|z| \leq c \cdot n(|z|)^c$  and  $n(|z|) \leq c' \cdot |z|^{c'}$ .

► **Definition 3** (Black-box reduction to distinguishing AIF). *Let  $L$  be a language and  $G := \{G_z : \{0,1\}^n \rightarrow \{0,1\}^{\ell(n)}\}_{z \in \{0,1\}^*}$  be an auxiliary-input function with  $\ell(n) > n$ . We say that there exists a black-box (BB) reduction from  $L$  to distinguishing  $G$  if for all polynomials  $p$ , there exists a randomized polynomial-time oracle machine  $R^?$  such that for all oracles  $\mathcal{O}$  that  $(1/p)$ -distinguish  $G$  and  $x \in \{0,1\}^*$ ,  $R$  satisfies that*

$$\Pr_R[R^{\mathcal{O}}(x) = L(x)] \geq 2/3.$$

Moreover, we say that there exists a nonadaptive BB reduction from  $L$  to distinguishing  $G$  if all  $R$  make their queries independently of any answer by oracle for previous queries.

The first main result on AIPRG is stated as follows.

► **Theorem 4.** *For any auxiliary-input function  $G = \{G_z : \{0,1\}^n \rightarrow \{0,1\}^{\ell(n)}\}_{z \in \{0,1\}^*}$  with  $\ell(n) > n$ , there exists no nonadaptive BB reduction from an NP-hard language  $L$  to distinguishing  $G$  unless the polynomial hierarchy collapses.*

## 2.2 Auxiliary-Input One-Way Function

A one-way function is a function which is easy to compute but hard to invert, and it is a fundamental primitive in the sense that most cryptographic tools do not exist without a one-way function [23, 32]. The formal definition is the following:

► **Definition 5** (One-way function). *Let  $s, \ell$  be polynomials. We say that a family of function  $f = \{f_n\}_{n \in \mathbb{N}}$  where  $f_n : \{0,1\}^{s(n)} \rightarrow \{0,1\}^{\ell(n)}$  is an (i.o.-)one-way function (OWF)<sup>2</sup> if  $f$  is polynomial-time computable, and there exists a polynomial  $p$  such that for all polynomial-time randomized algorithms  $A$ , there exist infinitely many  $n \in \mathbb{N}$  such that*

$$\Pr_{A, U_{s(n)}} [A(1^n, f_n(U_{s(n)})) \notin f_n^{-1}(f_n(U_{s(n)}))] \geq 1/p(n).$$

For the sake of simplicity, we may omit to write the input  $1^n$  to  $A$ .

The auxiliary-input analog of OWF, first introduced by Ostrovsky and Wigderson [30], is defined as follows.

► **Definition 6** (Auxiliary-input one-way function). *Let  $f = \{f_z : \{0,1\}^n \rightarrow \{0,1\}^{\ell}\}_{z \in \{0,1\}^*}$  be an auxiliary-input function and  $\gamma : \mathbb{N} \rightarrow (0,1)$  be a function. We say that a randomized algorithm  $A$   $\gamma$ -inverts  $f$  if for all  $z \in \{0,1\}^*$ ,*

$$\Pr_{A, U_n} [A(z, f_z(U_n)) \in f_z^{-1}(f_z(U_n))] \geq \gamma(n).$$

We say that  $f$  is an auxiliary-input one-way function (AIOWF) if there exists a polynomial  $p$  such that no polynomial-time randomized algorithm  $(1 - 1/p)$ -inverts  $f$ .

In fact, the existence of AIOWF and AIPRG is equivalent [15]. However, we cannot directly apply Theorem 4 to AIOWF due to the adaptive security reduction, as we mentioned in Section 1.1.

A BB reduction for AIOWF is defined as follows. It is easily verified that for any polynomial  $p$ , the following BB reduction from a language  $L$  to  $(1 - 1/p)$ -inverting an AIF  $f$  shows that  $f$  is an AIOWF if  $L \notin \text{BPP}$ .

<sup>2</sup> Strictly speaking, a one-way function defined in Definition 5 is usually called a “weak” one-way function, which implies the standard (strong) one-way function.

► **Definition 7** (Black-box reduction to inverting AIF). *Let  $L$  be a language,  $p$  be a polynomial, and  $f := \{f_z : \{0,1\}^n \rightarrow \{0,1\}^\ell\}_{z \in \{0,1\}^*}$  be an auxiliary-input function. We say that a randomized polynomial-time oracle machine  $R^?$  is a black-box (BB) reduction from  $L$  to  $(1 - 1/p)$ -inverting  $f$  if for all oracles  $\mathcal{O}$  that  $(1 - 1/p)$ -invert  $f$  and  $x \in \{0,1\}^*$ ,  $R$  satisfies that*

$$\Pr_R[R^{\mathcal{O}}(x) = L(x)] \geq 2/3.$$

Moreover, we say that  $R$  is nonadaptive if all  $R$ 's queries are made independently of any answer by oracle for previous queries.

The second main result on AIOWF is stated as follows.

► **Theorem 8.** *For any auxiliary-input function  $f = \{f_z : \{0,1\}^n \rightarrow \{0,1\}^\ell\}_{z \in \{0,1\}^*}$  and polynomial  $p$ , if there exists a nonadaptive BB reduction from an NP-hard language  $L$  to  $(1 - 1/p)$ -inverting  $f$ , then  $\text{NP} \not\subseteq \text{BPP}$  also implies that a one-way function exists (via an adaptive BB reduction).*

## 2.3 Auxiliary-Input Hitting Set Generator

A hitting set generator is a weak variant of a pseudorandom generator, introduced in the context of derandomization by Andreev et al. [4]. For the original purpose, they considered (possibly) exponential-time computable generators. In this paper, we focus on polynomial-time computable generators as in cryptography. We define the auxiliary-input analog as follows.

► **Definition 9** (Auxiliary-input hitting set generator). *Let  $G = \{G_z : \{0,1\}^n \rightarrow \{0,1\}^{\ell(n)}\}_{z \in \{0,1\}^*}$  be an auxiliary-input function. For a function  $\gamma : \mathbb{N} \rightarrow (0,1)$ , we say that a randomized adversary  $A$   $\gamma$ -avoids  $G$  if for all (public) auxiliary-inputs  $z \in \{0,1\}^*$  and (private) inputs  $x \in \{0,1\}^{n(|z|)}$ ,*

$$\Pr_A[A(z, G_z(x)) = 0] \geq 2/3 \quad \text{and} \quad \Pr_{y \sim \{0,1\}^{\ell(n(|z|))}} \left[ \Pr_A[A(z, y) = 1] \geq 2/3 \right] \geq \min(\gamma(n), \tau_z),$$

where  $\tau_z$  be a trivial limitation<sup>3</sup> defined as  $\tau_z = 1 - \frac{|G_z(\{0,1\}^n)|}{2^{\ell(n)}}$ .

We say that  $G$  is a  $\gamma$ -secure auxiliary-input hitting set generator (AIHSG) if  $\ell(n) > n$  and there exists no polynomial-time randomized algorithm  $(1 - \gamma)$ -avoiding  $G$ .

Although it is easily verified that AIPRG is also AIHSG (for any security  $\gamma(n) = 1/\text{poly}(n)$ ), the opposite direction is open at present. In fact, the hardness of learning implies the existence of AIHSG [28]; on the other hand, we must overcome the barrier by oracle separation to show the existence of AIPRG (equivalently, AIOWF) from the hardness of learning [35]. Thus, AIHSG seems to be a much weaker notion than AIOWF and AIPRG under current knowledge.

A BB reduction for AIHSG is defined as follows. It is easily verified that the following BB reduction from a language  $L$  to  $(1 - \gamma)$ -avoiding an AIF  $G$  shows that  $G$  is a  $\gamma$ -secure AIHSG if  $L \notin \text{BPP}$ .

<sup>3</sup> In this paper, we consider general settings of  $\gamma$  and  $\ell$ . Thus, we adopted the trivial limitation in the definition to avoid arguing about invalid settings where  $\gamma$ -avoiding the generator is impossible by definition.



► **Definition 10** (Black-box reduction to avoiding AIF). *Let  $L$  be a language,  $\gamma$  be a function, and  $G := \{G_z : \{0,1\}^n \rightarrow \{0,1\}^\ell\}_{z \in \{0,1\}^*}$  be an auxiliary-input function. We say that a randomized polynomial-time oracle machine  $R^?$  is a black-box (BB) reduction from  $L$  to  $(1 - \gamma)$ -avoiding  $G$  if for all oracles  $\mathcal{O}$  that  $(1 - \gamma)$ -avoid  $G$  and  $x \in \{0,1\}^*$ ,  $R$  satisfies that*

$$\Pr_R[R^{\mathcal{O}}(x) = L(x)] \geq 2/3.$$

*Moreover, we say that  $R$  is nonadaptive if all  $R$ 's queries are made independently of any answer by oracle for previous queries.*

The third main result on AIHSG is stated as follows.

► **Theorem 11.** *Let  $p$  be a polynomial and  $G := \{G_z : \{0,1\}^n \rightarrow \{0,1\}^{\ell(n)}\}_{z \in \{0,1\}^*}$  be an auxiliary-input function where  $\ell(n) > (1 + \epsilon) \cdot n$  for some constant  $\epsilon > 0$ . If there exists a nonadaptive BB reduction from an NP-hard language  $L$  to  $(1 - 1/p)$ -avoiding  $G$ , then  $\text{NP} \not\subseteq \text{BPP}$  also implies that a one-way function exists (via an adaptive BB reduction).*

### 3 Discussion and Future Directions

As discussed in Section 1.1, Theorems 8 and 11 are also regarded as approaches to construct one-way functions based on NP-hardness. In this section, we discuss the novelty of this optimistic perspective and suggest future directions, including the investigation of the validity.

Our results are rephrased as follows: Assume that we could connect NP-hardness to some auxiliary-input primitives (i.e., AIOWF or AIHSG) via a novel nonadaptive BB reduction, then we can automatically extend the connection to standard cryptographic primitives, that is, OWF. At present, the latter task of removing auxiliary-input from primitives seems quite non-trivial, as mentioned in [5, 33]. In this paper, we also provide a simple oracle separation between AIOWF and OWF as follows. This indicates that we cannot expect any relativized technique to remove auxiliary-input from cryptographic primitives.

► **Theorem 12.** *There exists an oracle  $\mathcal{O}$  such that relative to  $\mathcal{O}$  an auxiliary-input one-way function exists, but a one-way function does not exist.*

Additionally, there are several barriers by other oracle separations at the intermediate levels to base OWF on NP-hardness (e.g., [35, 21]). Although such barriers on relativization are common throughout theoretical computer science (e.g., the P vs. NP problem [6]), there are only a few success stories of overcoming such barriers at present. Unfortunately, Theorems 8 and 11 do not provide any solution to break these barriers, and a new non-relativized technique is still required. Specifically, if a nonadaptive BB reduction to AIOWF or AIHSG is also relativized<sup>4</sup>, then our proof also yields relativized reductions that contradict Theorem 12 or the oracle separation presented in [21].

However, our result offers one hope. Although there seems to be several barriers towards cryptography based on NP-hardness as discussed above, the essential barrier we must overcome might be few. Theorems 8 to 12 certainly show that if we could find a non-relativized breakthrough at an intermediate level toward cryptography (that is, auxiliary-input primitives), then it will be lifted and break the other barriers at the higher level. From this perspective, we conjecture that the difficulty in basing OWF on NP-hardness could rely on

<sup>4</sup> Note that oracle separations do not necessarily rule out BB reductions from particular languages, not as fully BB reductions defined in [31].

a much smaller part of tasks at an intermediate level. This conjecture seems somewhat controversial but enhances the significance of further investigation on auxiliary-input or other intermediate cryptographic primitives instead of standard ones.

The above discussion leads to the following two possible directions. The first direction is to find other scenarios where a breakthrough at an intermediate level also brings benefits at the higher level. This direction might reduce constructing standard cryptographic primitives to the task at the low level and give new insights into complexity-based cryptography. The second direction is to refute such an attempt on intermediate primitives with convincing evidence if it gives the wrong direction. Particularly, in our case, there is a possibility that nonadaptive BB reductions for AIOWF and AIHSG indeed yield the collapse of the polynomial-hierarchy as in the case of AIPRG.

For the second direction, we list two concrete ways: (1) finding a new construction of AIPRG from AIOWF with nonadaptive security proof; (2) generalizing the previous results for OWF [2] or HSG [17] to each auxiliary-input analog for the stronger consequence. At least the latter approach seems to require some new technique to simulate nonadaptive BB reductions by constant-round interactive proof systems, as observed in [5] and [34].

#### 4 A First Attempt: Applying [10] and [2]

Before presenting our proof strategies, we roughly explain why the previous technique developed by Bogdanov and Trevisan [10] for the worst-to-average framework is not applicable in the case of auxiliary-input cryptography. For the sake of simplicity, we assume that there exists a nonadaptive BB reduction  $R$  from an NP-hard language  $L$  to a distributional NP-problem  $(L', U)$ , and  $R$  makes queries of the same length  $n$  determined by the size of input to  $R$ . Note that if we can answer these queries by an oracle which correctly recognizes  $L'$  on average, then  $R$  must recognize  $L$ . Bogdanov and Trevisan construct an AM/poly protocol for recognizing  $L$  by leaving this role of the oracle to a prover, which implies that  $\text{coNP} \subseteq \text{AM/poly}$  and the collapse of the polynomial hierarchy.

Roughly speaking, their central idea is to divide each  $R$ 's query  $x \in \{0, 1\}^n$  into “light” and “heavy” queries according to the probability  $p_x$  that the query  $x$  is generated by  $R$ . Specifically, they determine a threshold  $p(n) = \text{poly}(n)$  depending on the permissible error probability for solving  $(L', U)$  on average and define a light (resp. heavy) query  $x$  as a query satisfying the condition  $p_x \leq p(n)2^{-n}$  (resp.  $p_x > p(n)2^{-n}$ ). Then, they make the prover answer (ideally) all light queries correctly, i.e., simulate the following oracle.

$$\mathcal{O}_{\mathcal{L}} = \{x \in \{0, 1\}^* : x \in L' \text{ and } x \text{ is a light query}\}$$

Because the number of heavy queries is at most  $2^n/p(n)$ , the above oracle  $\mathcal{O}_{\mathcal{L}}$  solves  $(L', U)$  with error probability at most  $p(n)$ . Therefore, it is enough to make a prover simulate  $\mathcal{O}_{\mathcal{L}}$  for constructing an AM/poly protocol which recognizes  $L$  based on  $R$ . For the soundness, the verifier must accomplish the following two tasks without being deceived by malicious provers: (1) distinguishing between light and heavy queries and (2) identifying the correct answer for each light query. Bogdanov and Trevisan developed such a verifier by introducing sophisticated protocols called the heavy sampling protocol and the hiding protocol.

Herein, we consider the case of auxiliary-input primitives, where each  $R$ 's query takes the form of  $(z, x)$  where  $z$  denotes auxiliary-input. For the sake of simplicity, we assume that the task of breaking an auxiliary-input primitive is further reduced to an average-case deterministic problem on uniform distribution with auxiliary-input by applying the techniques

in [22, 7] as in the previous work [10] and the length of instances of the average-case problem is the same as the length of auxiliary-input. We also assume that a reduction  $R$  makes queries of the form  $(z, x)$  where  $|z| = |x| = n$  and  $n$  is determined by the size of input to  $R$ .

There are two possible ways to extend the above idea to the case of auxiliary-input primitives: considering auxiliary-input in queries (a) together or (b) separately. For the first approach (a), we must determine a light query as a query satisfying the condition  $p_x \leq \text{poly}(n)2^{-2n}$  for applying the hiding protocol. This is problematic because the number of heavy queries is possibly  $2^{2n}/\text{poly}(n)$ , and many of them can be concentrated on one auxiliary-input. In other words, the above oracle  $\mathcal{O}_{\mathcal{L}}$  does not always solve the average-case problem in the worst-case sense on auxiliary-input. On the other hand, for the second approach (b), their verifier needs information on some statistics as advice for each auxiliary-input. Because there are exponentially many possibilities on auxiliary-input, the total length of such advice is exponentially large, which is unfeasible as an  $\text{AM}/\text{poly}$  protocol.

The subsequent work [2] provided the method to remove the above advice in the case of standard cryptographic primitives by applying an additional property of breaking cryptographic primitives (therefore, they constructed an  $\text{AM}$  protocol for  $\neg L$  instead of an  $\text{AM}/\text{poly}$  protocol). Unfortunately, even this method cannot be applied directly in the case of auxiliary-input cryptography. To obtain the statistics corresponding to the above advice, their protocol needs to generate query set by executing  $R$ . In our case, remember that we consider each auxiliary-input separately, so we need to simulate a conditional distribution on queries for fixed auxiliary-input. However, such distributions are not efficiently samplable in general: for example, consider the query distribution on  $(h(y), y)$  where  $h$  is a collision-free hashing function. Then, a polynomial-time verifier which simulates a conditional distribution for a fixed auxiliary-input (i.e., hash value) can easily find the collision of  $h$ .

## 5 Proof Sketches

In this section, we present proof ideas of Theorems 4, 8, 11, and 12. Note that Theorem 11 heavily relies on Theorem 8, and Theorem 8 heavily relies on Theorem 4. Therefore, although each proof idea may look pretty simple and intuitive, our construction of OWF for Theorem 11 becomes complicated and quite non-trivial as a whole. For the formal proofs, refer to the full version [29].

### 5.1 The Case of AIPRG: Proof Idea of Theorem 4

First, we formally introduce a hitting set generator, which takes a crucial role in our proof.

► **Definition 13** (Hitting set generator). *Let  $\gamma(n)$  be a function. A function  $G : \{0, 1\}^n \rightarrow \{0, 1\}^{\ell(n)}$  with  $\ell(n) > n$  is a (polynomial-time computable)  $\gamma$ -secure hitting set generator (HSG) if  $G$  is polynomially computable and there is no polynomial-time randomized adversary  $A$   $\gamma$ -avoiding  $G$ , i.e., satisfying the condition that for all sufficiently large  $n \in \mathbb{N}$ ,*

$$\forall x \in \{0, 1\}^n \Pr_A[A(G(x)) = 0] \geq 2/3 \text{ and } \Pr_{y \sim \{0, 1\}^{\ell(n)}} \left[ \Pr_A[A(y) = 1] \geq 2/3 \right] \geq \min(\gamma(n), \tau_n),$$

where  $\tau_n$  be a trivial limitation defined as  $\tau_n := 1 - \frac{|G(\{0, 1\}^n)|}{2^{\ell(n)}}$ .

Theorem 4 essentially follows from a nonadaptive BB security reduction from distinguishing AIPRG to avoiding HSG. Note that HSG based on AIPRG with a nonadaptive BB security reduction has been implicitly given in the study on MCSP [3, 16]. To see this explicitly, we will provide a much simpler construction of HSG based on AIPRG and a

self-contained proof. Although the reader may think that our construction is too fundamental and looks somewhat trivial, to the best of our knowledge, no one has mentioned such a direct relationship between AIPRG and HSG.

First, we assume that there is a nonadaptive BB security reduction from distinguishing AIPRG to avoiding HSG. Avoiding HSG is directly formulated as the following distributional NP problem (with zero-error): for uniformly chosen  $y$ , determine whether  $y$  is contained in the image of HSG. Therefore, the reduction also yields a nonadaptive BB reduction from distinguishing AIPRG to the distributional NP problem. Thus, any nonadaptive BB reduction from an NP-hard problem to distinguishing AIPRG indeed yields a nonadaptive BB reduction from the same NP-hard problem to the distributional NP problem. By the previous result by Bogdanov and Trevisan [10], such a reduction implies the collapse of the polynomial-hierarchy.

Our construction of HSG from AIPRG is the following: just considering the both of auxiliary-input and input to AIPRG as usual input to HSG. More specifically, let  $G = \{G_z : \{0, 1\}^n \rightarrow \{0, 1\}^{\ell(n)}\}_{z \in \{0, 1\}^*}$  be an AIPRG. Then the construction of HSG  $G'$  is given as  $G'(z \circ x) = G_z(x)$ . Note that, when  $z + n(|z|) \geq \ell(n(|z|))$  holds,  $G'$  does not satisfy the syntax on stretching input. In the formal proof, therefore, we first stretch the output of  $G$  by the standard technique in cryptography. It can be easily verified that the security reduction for this stretching (shown by the famous hybrid argument) is nonadaptive.

Let  $\gamma(n)$  be a reciprocal of polynomial. The security reduction from  $\gamma$ -avoiding  $G'$  to distinguishing  $G$  is also simple: just employing an adversary  $A$  for  $G'$  as an adversary for  $G$ . Obviously, this reduction is nonadaptive. To show the correctness, assume that  $A$   $\gamma$ -avoids  $G'$ . For the sake of simplicity, we also assume that  $A$  is deterministic and  $\gamma(n) < \tau_n$ . Whenever the input  $y$  is pseudorandom string contained in the image of  $G'$ ,  $A(y)$  does not output 1. On the other hand, if  $y$  is a truly random string, then  $A(y)$  outputs 1 with probability at least  $\gamma(n)$ . Thus,  $A$  can distinguish the uniform distribution from all distributions on the image of  $G'$  with an advantage at least  $\gamma(n)$ . For any auxiliary-input  $z$ ,  $G_z(U_{n(|z|)})$  is distributed on the image of  $G'$ . Thus,  $A$  also  $\gamma$ -distinguishes  $G$ .

## 5.2 The Case of AIOWF: Proof Idea of Theorem 8

In this section, we omit all arguments about the success probabilities of adversaries to focus on the proof idea. First, we introduce several reductions as elements of a standard OWF. Let  $R_{L \rightarrow f}$  denote the nonadaptive BB reduction from  $L$  to inverting  $f$  in the assumption. By the construction of PRG from OWF (e.g., [15]), there exist an auxiliary-input generator  $G$  and an adaptive BB reduction  $R_{f \rightarrow G}$  from inverting  $f$  to distinguishing  $G$ . By the result in Section 5.1, there exist an NP-language  $L'$  and a nonadaptive BB reduction  $R_{G \rightarrow L'}$  from distinguishing  $G$  to a distributional NP problem  $(L', U)$  (with zero-error). Since  $L' \in \text{NP}$  and  $L$  is NP-hard, there exists a Karp reduction  $R_{L' \rightarrow L}$  from  $L'$  to  $L$ .

Now we consider the following procedure:

1. select an instance  $x'$  of  $L'$  at random;
  2. translate  $x'$  into an instance  $x$  of  $L$  as  $x = R_{L' \rightarrow L}(x')$ ;
  3. plug  $x$  into  $R_{L \rightarrow f}$  with a random tape  $r$ ;
- At this stage,  $R_{L \rightarrow f}$  makes polynomially many queries  $(z_1, y_1), \dots, (z_q, y_q)$ .
4. answer the queries by some inverting oracle  $\mathcal{O}$ ;
  5. if  $R_{L \rightarrow f}$  outputs  $b \in \{0, 1\}$ , then output the same decision  $b$ .

Note that if the oracle  $\mathcal{O}$  correctly inverts  $f$ , then the resulting decision  $b$  is  $L(x)$  with high probability by the property of  $R_{L \rightarrow f}$ , and  $L(x)$  is equal to  $L'(x')$  by the property of  $R_{L' \rightarrow L}$ .

The crucial observation is that there is no worst-case sense at all in the above procedure because both  $x'$  and  $r$  are selected at random. Therefore, all queries at the stage 3 are indeed efficiently samplable, and the inverting oracle no longer needs to invert  $f$  for every auxiliary-input at the stage 4. This observation leads to the following construction of a standard OWF  $g$ .

The function  $g$  takes three inputs  $x', r$ , and  $x^f$ , which intuitively represents a random instance of  $L'$ , randomness for  $R_{L \rightarrow f}$ , and input for  $f$ , respectively. Then  $g(x', r, x^f)$  imitates the above procedure as follows: (2') translate  $x'$  into an instance  $x$  of  $L$  as  $x = R_{L' \rightarrow L}(x')$ , (3') plug  $x$  into  $R_{L \rightarrow f}$  with randomness  $r$ , then randomly pick one of auxiliary-input  $z$  in queries by  $R_{L \rightarrow f}$  and output  $f_z(x^f)$ .

We will show that the above  $g$  is one-way if  $\text{NP} \not\subseteq \text{BPP}$ . For contradiction, we assume that there exists an adversary  $A$  that inverts  $g$ . Remember that  $g$  simulates a distribution on queries produced by  $R_{L \rightarrow f}$  in the above procedure. Thus, intuitively, we can replace the inverting oracle  $\mathcal{O}$  with the adversary  $A$  at the stage 4 with high probability. This is a little technical part, and we present further details in the full version [29]. Then the above procedure no longer needs any oracle and yields a randomized algorithm solving  $(L', U)$  on average. By applying reductions  $R_{G \rightarrow L'}$ ,  $R_{f \rightarrow G}$ , and  $R_{L \rightarrow f}$  in this order, this also yields a randomized polynomial-time algorithm for  $L$ . Since  $L$  is NP-hard, we conclude that  $\text{NP} \subseteq \text{BPP}$ .

Remark that  $R_{G \rightarrow L'}$  is a nonadaptive BB reduction thanks to our simple construction in Section 5.1. Therefore, if we also have a construction of AIPRG  $G$  from AIOWF  $f$  with a nonadaptive BB reduction from inverting  $f$  to distinguishing  $G$ , then the above proof leads to a nonadaptive BB reduction from  $L$  to  $(L', U)$ , which implies the collapse of the polynomial hierarchy as in Theorem 4. Thus, finding such a simple construction of AIPRG is one direction for excluding a nonadaptive BB reduction to base AIOWF on NP-hardness, as mentioned in Section 3.

### 5.3 The Case of AIHSG: Proof Idea of Theorem 11

Our goal is to simulate an avoiding oracle for a nonadaptive BB reduction by another protocol in some restricted complexity class, in our case, BPP. The key idea for this is to classify each query generated by the nonadaptive BB reduction into “light” and “heavy” queries as in [10]. A similar technique was also applied in the previous work for HSG [12, 17]. Thus, we first review the previous case of HSG and then explain the difference to our case of AIHSG.

#### The Case of Hitting Set Generator (Previous work)

Let  $G : \{0, 1\}^n \rightarrow \{0, 1\}^{\ell(n)}$  denote a generator with  $\ell(n) \geq (1 + \Omega(1)) \cdot n$  and  $R^?$  denote a nonadaptive BB reduction from an NP-language  $L$  to avoiding  $G$ . W.l.o.g., we can assume that marginal distributions on each query by  $R$  are identical regardless of each query position by applying a random permutation on query positions before asking them to oracle. Thus, for each input  $x \in \{0, 1\}^n$  to  $R$ , one marginal distribution  $Q_x$  on  $R$ 's queries is determined. We choose a threshold (roughly)  $\tau = 1/\tilde{\Theta}(2^n)$  and define a light (resp. heavy) query  $y \in \{0, 1\}^{\ell(n)}$  as a query generated according to  $Q_x$  with probability less (resp. greater) than the threshold  $\tau$ .

We simulate the avoiding oracle for  $G$  by using the classification of queries as follows. First, assume that we could (somehow) distinguish the heavy case and the light case for a given query. Then we can also simulate one of avoiding oracles simply as follows: for each query  $y$  generated by  $R(x)$ , (1) determine whether  $y$  is heavy or light; (2) answer 0 (resp. 1)

if  $y$  is heavy (resp. light) query. Let  $\mathcal{O}'$  denote the induced oracle by the above simulating procedure. Note that the probability that  $\mathcal{O}'(y)$  outputs 0 is exponentially small because the fraction of light queries is  $\tilde{\Theta}(2^n)/2^{\ell(n)} \leq 2^{-\Omega(n)}$ . Thus,  $\mathcal{O}'$  satisfies the condition on the probability of outputting 1. However,  $\mathcal{O}'$  is not avoiding oracle for  $G$ , because there is possibly a query  $y$  such that  $y$  is heavy but contained in  $\text{Im}G$ . In this case,  $\mathcal{O}'(y)$  outputs 1 even for  $y \in \text{Im}G$  and fails to avoid  $G$ .

The key observation to overcome this issue is the following:

- ( $\star$ ) For each length  $\ell(n)$  of query (i.e., the input size is  $n$ ), the size of  $\text{Im}G$  is at most  $2^n$ ; thus the probability that  $R$  asks some light query contained in  $\text{Im}G$  (we refer to it as a “bad” query) is bounded above by  $2^n/\tilde{\Theta}(2^n) \leq 1/\text{poly}(n)$ .

Therefore,  $\mathcal{O}'$  is consistent with some avoiding oracle, and  $R^{\mathcal{O}'}(x)$  correctly recognizes  $x$  with high probability over the execution of  $R$ .

By the above argument, we can reduce avoiding a generator to distinguishing heavy and light queries. For the latter task, Gutfreund and Vadhan [12] presented a  $\text{BPP}^{\text{NP}}$  algorithm by approximation of counting in [26], and Hirahara and Watanabe [17] presented an  $\text{AM} \cap \text{coAM}$  algorithm by generalizing the protocol in [11].

### The Case of Auxiliary-input Hitting Set Generator (Our work)

We move on to our case of AIHSG. Let  $G = \{G_z : \{0, 1\}^{n(|z|)} \rightarrow \{0, 1\}^{\ell(n(|z|))}\}_{z \in \{0, 1\}^*}$  denote an auxiliary-input generator with  $\ell(n) \geq (1 + \Omega(1)) \cdot n$  and  $R^?$  denote a nonadaptive BB reduction from an NP-language  $L$  to avoiding  $G$ . We can also assume that all marginal query distributions of  $R^?(x)$  are identical to  $Q_x$  regardless of query position.

To extend the above argument to our case of AIHSG, the problematic part is the key observation ( $\star$ ). Remember that an adversary for AIHSG must avoid  $G_z$  for all  $z \in \{0, 1\}^*$ , and auxiliary-input is possibly longer than output. Therefore, we cannot bound the size of the image of the generator in general because the image may span the whole range (for example, consider the following generator  $G_z(x) = z \oplus (x \circ 0^{|z|-|x|})$  for  $|z| > n(|z|)$ ).

To overcome this, we need to consider each case of auxiliary-input  $z$  separately. Therefore, we change the definitions of “light” and “heavy” queries depending on auxiliary-input. Let  $p_x(z)$  denote a probability that  $Q_x$  generates a query of auxiliary-input  $z$ . If we can bound the probability that  $R$  makes light query  $(z, y)$  with  $y \in \text{Im}G_z$  by  $1/(\text{poly}(n) \cdot p_x(z))$  for any  $z$ , then  $R$  makes such a “bad” query  $(z, y)$  with probability at most  $\sum_z p_x(z) \cdot 1/(\text{poly}(n) \cdot p_x(z)) = 1/\text{poly}(n)$ . Then we can use the same argument in the case of HSG and reduce avoiding  $G$  to distinguishing heavy and light cases. This idea naturally leads to the following new definition of “light” and “heavy”: separating each query  $(z, y)$  by the conditional probability  $p_x(y|z)$  that  $y$  is asked conditioned on the event that the auxiliary-input in the query is  $z$ . In fact, this modification will work well even for AIHSG (for the formal argument, refer to the full version [29]).

However, one issue remains: how can we distinguish heavy and light queries? To this end, we must verify the largeness of the conditional probability of the given query. This part essentially prevents us from applying the previous results. Since we consider a polynomial-time computable generator, the simulation with NP oracle does not yield any nontrivial result, not as the work in [12]<sup>5</sup>. Even for the simulation in  $\text{AM} \cap \text{coAM}$  in [17], there are

<sup>5</sup> Their work concerned the original aim of HSG, i.e., derandomization (e.g., [25]). For this purpose, they considered (possibly) exponential-time computable HSG  $G$ , where avoiding  $G$  in  $\text{BPP}^{\text{NP}}$  is quite nontrivial. However, in our case where  $G$  is polynomial-time computable, avoiding  $G$  is in NP trivially.



several technical issues. We cannot trivially verify the size of conditional probability by such protocols due to the restricted use of the upper bound protocol developed in [1]. Moreover, we cannot possibly even sample the conditional distribution efficiently for fixed auxiliary-input, as discussed in Section 4.

Our idea is to adopt universal extrapolation in [22]. Intuitively speaking, the universal extrapolation is a tool to reduce approximating the probability  $p_y = \Pr_{U_n}[y = f(U_n)]$  to inverting  $f$  for a polynomial-time computable  $f$  and a given  $y = f(x)$  where  $x \in \{0, 1\}^n$  is selected at random. In fact, the universal extrapolation holds even for an auxiliary-input function, and a similar technique was also used in [30]. By using the universal extrapolation for each circuit which generates query and auxiliary-input, we have a good approximation of  $p_x(y|z)$  for query  $(z, y)$  generated by  $R^2(x)$ . Thus, the universal extrapolation enables us to classify the given  $(z, y)$  correctly. Note that the auxiliary-input in the universal extrapolation essentially corresponds to the input  $x$  for each circuit sampling query and auxiliary-input.

To show Theorem 11, we need further observations. Since  $R$  makes its queries non-adaptively, we can also invoke the universal extrapolation nonadaptively. Moreover, the universal extrapolation algorithm indeed uses an inverting adversary for a certain AIOWF as black-box and nonadaptively (we also see this formally in the full version [29]). As a result, a nonadaptive BB reduction from an NP-hard language  $L$  to avoiding AIHSG yields a nonadaptive BB reduction from  $L$  to inverting AIOWF. Thus, by Theorem 8,  $R$  also yields a one-way function under the assumption that  $\text{NP} \not\subseteq \text{BPP}$ .

## 5.4 Oracle Separation between OWF and AIOWF: Proof Idea of Theorem 12

To show Theorem 12, we employ a random function  $\mathcal{F} = \{\mathcal{F}_n : \{0, 1\}^n \rightarrow \{0, 1\}^n\}_{n \in \mathbb{N}}$ , where each  $\mathcal{F}_n$  is selected uniformly from length-preserving functions of input size  $n$ . As shown in [24], any polynomial-time oracle machine cannot invert  $\mathcal{F}$  with non-negligible probability (with probability 1 over the choice of  $\mathcal{F}$ ). In other words, if a primitive given access to  $\mathcal{F}$  directly outputs the value of  $\mathcal{F}$ , such a primitive must be one-way. Therefore, all we have to do is to let a random function  $\mathcal{F}$  available for auxiliary-input primitives but unavailable for standard primitives.

To this end, we simply add  $n$ -bit auxiliary-input to a random function of the input size  $n$ . Then we choose one auxiliary-input  $z_n$  from  $2^n$  possibilities of  $\{0, 1\}^n$  as a target auxiliary-input and embed the random function to the position indexed by  $z_n$ . Let  $\mathcal{F} = \{F_z : \{0, 1\}^{|z|} \rightarrow \{0, 1\}^{|z|}\}_{z \in \{0, 1\}^*}$  be such an embedded random function. Note that the similar random embedding technique was also used in the previous work for other oracle separations (e.g., [35]). If an auxiliary-input primitive  $f$  given access to  $\mathcal{F}$  identifies the auxiliary-input of  $F$  with own auxiliary-input, then  $f$  must be AIOWF because an adversary for  $f$  must invert  $f_z$  for all auxiliary-inputs  $z$ , including the random function. On the other hand, any polynomial-time computable primitive (without auxiliary-input) cannot find the target auxiliary-input of  $\mathcal{F}$  with non-negligible probability because they were selected at random. Thus, any (standard) primitive does not take nontrivial advantage of  $\mathcal{F}$ .

For the oracle separation, we combine the above embedded random function  $\mathcal{F}$  with the PSPACE oracle (w.l.o.g., the oracle TQBF determining satisfiability of quantified Boolean formulae). Let  $\mathcal{O}_{\mathcal{F}}$  denote this oracle. Since the random function in  $\mathcal{F}$  is selected independently of TQBF, the additional access to TQBF does not help to invert the random function at all. Thus, AIOWF still exists relative to  $\mathcal{O}_{\mathcal{F}}$ .

On the other hand, we consider a function  $f$  which is polynomial-time computable with access to  $\mathcal{O}_{\mathcal{F}}$  arbitrarily. Since the target auxiliary-input is selected independently of TQBF, the additional access to TQBF does not help to find the target auxiliary-input at all. Thus,  $f$



cannot still take nontrivial advantage of  $\mathcal{F}$  and is regarded as a function given only access to TQBF. We can easily verify that any polynomial-time computable function with access to TQBF is efficiently invertible by TQBF. Since the above argument holds for any  $f$ , OWF does not exist relative to  $\mathcal{O}_{\mathcal{F}}$ . Thus, we have the oracle separation between AIOWF and OWF.

---

## References

- 1 W. Aiello and J. Håstad. Perfect zero-knowledge languages can be recognized in two rounds. In *28th Annual Symposium on Foundations of Computer Science*, pages 439–448, 1987.
- 2 A. Akavia, O. Goldreich, S. Goldwasser, and D. Moshkovitz. On Basing One-Way Functions on NP-Hardness. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, STOC '06, pages 701–710, New York, NY, USA, 2006. ACM.
- 3 E. Allender and S. Hirahara. New Insights on the (Non-)Hardness of Circuit Minimization and Related Problems. *TOCT*, 11(4):27:1–27:27, 2019.
- 4 A. Andreev, A. Clementi, and J. Rolim. A New General Derandomization Method. *J. ACM*, 45(1):179–213, January 1998.
- 5 B. Applebaum, B. Barak, and D. Xiao. On Basing Lower-Bounds for Learning on Worst-Case Assumptions. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science*, FOCS'08, pages 211–220, 2008.
- 6 T. Baker, J. Gill, and R. Solovay. Relativizations of the  $\mathcal{P} = ? \mathcal{NP}$  Question. *SIAM Journal on Computing*, 4(4):431–442, 1975.
- 7 S. Ben-David, B. Chor, O. Goldreich, and M. Luby. On the theory of average case complexity. *Journal of Computer and System Sciences*, 44(2):193–219, 1992. doi:10.1016/0022-0000(92)90019-F.
- 8 A. Bogdanov and C. Brzuska. On Basing Size-Verifiable One-Way Functions on NP-Hardness. In *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part I*, pages 1–6, 2015.
- 9 A. Bogdanov and C. Lee. Limits of Provable Security for Homomorphic Encryption. In *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I*, pages 111–128, 2013.
- 10 A. Bogdanov and L. Trevisan. On Worst-Case to Average-Case Reductions for NP Problems. *SIAM J. Comput.*, 36(4):1119–1159, December 2006.
- 11 J. Feigenbaum and L. Fortnow. On the random-self-reducibility of complete sets. In *Proceedings of the 6th Annual Structure in Complexity Theory Conference*, pages 124–132, 1991.
- 12 D. Gutfreund and S. Vadhan. Limitations of Hardness vs. Randomness under Uniform Reductions. In *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques. APPROX 2008, RANDOM 2008*, volume 5171 of *LNCS*, pages 469–482, 2008.
- 13 I. Haitner, M. Mahmoody, and D. Xiao. A New Sampling Protocol and Applications to Basing Cryptographic Primitives on the Hardness of NP. In *IEEE 25th Annual Conference on Computational Complexity*, pages 76–87, 2010.
- 14 I. Haitner, O. Reingold, and S. Vadhan. Efficiency Improvements in Constructing Pseudorandom Generators from One-Way Functions. *SIAM Journal on Computing*, 42(3):1405–1430, 2013.
- 15 J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby. A Pseudorandom Generator from Any One-way Function. *SIAM J. Comput.*, 28(4):1364–1396, March 1999.
- 16 S. Hirahara. Non-Black-Box Worst-Case to Average-Case Reductions within NP. In *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 247–258, 2018.
- 17 S. Hirahara and O. Watanabe. On Nonadaptive Reductions to the Set of Random Strings and Its Dense Subsets. In *Complexity and Approximation - In Memory of Ker-I Ko*, pages 67–79, 2020.

- 18 T. Holenstein. Key Agreement from Weak Bit Agreement. In *Proceedings of the Thirty-Seventh Annual ACM Symposium on Theory of Computing*, STOC '05, pages 664–673, New York, NY, USA, 2005. ACM.
- 19 T. Holenstein. Pseudorandom Generators from One-Way Functions: A Simple Construction for Any Hardness. In *Theory of Cryptography*, pages 443–461, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.
- 20 R. Impagliazzo. A personal view of average-case complexity. In *Proceedings of IEEE Tenth Annual Conference on Structure in Complexity Theory*, pages 134–147, 1995.
- 21 R. Impagliazzo. Relativized Separations of Worst-Case and Average-Case Complexities for NP. In *2011 IEEE 26th Annual Conference on Computational Complexity*, pages 104–114, 2011.
- 22 R. Impagliazzo and L. Levin. No better ways to generate hard NP instances than picking uniformly at random. In *Proceedings of the 31st Annual Symposium on Foundations of Computer Science*, FOCS'90, pages 812–821, 1990.
- 23 R. Impagliazzo and M. Luby. One-way Functions Are Essential for Complexity Based Cryptography. In *Proceedings of the 30th Annual Symposium on Foundations of Computer Science*, pages 230–235, 1989.
- 24 R. Impagliazzo and S. Rudich. Limits on the Provable Consequences of One-Way Permutations. In *Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing*, STOC '89, pages 44–61, New York, NY, USA, 1989. ACM.
- 25 R. Impagliazzo and A. Wigderson. Randomness vs Time: Derandomization under a Uniform Assumption. *Journal of Computer and System Sciences*, 63(4):672–688, 2001.
- 26 M. Jerrum, L. Valiant, and V. Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theoretical Computer Science*, 43:169–188, 1986.
- 27 T. Liu and V. Vaikuntanathan. On Basing Private Information Retrieval on NP-Hardness. In *Theory of Cryptography - 13th International Conference, TCC 2016-A, Tel Aviv, Israel, January 10-13, 2016, Proceedings, Part I*, pages 372–386, 2016.
- 28 M. Nanashima. Extending Learnability to Auxiliary-Input Cryptographic Primitives and Meta-PAC Learning. In *Proceedings of the 33rd Conference on Learning Theory, COLT'20*, volume 125, pages 2998–3029. PMLR, 09–12 July 2020.
- 29 M. Nanashima. On Basing Auxiliary-Input Cryptography on NP-hardness via Nonadaptive Black-Box Reductions. *Electron. Colloquium Comput. Complex.*, 27:95, 2020. URL: <https://eccc.weizmann.ac.il/report/2020/095>.
- 30 R. Ostrovsky and A. Wigderson. One-way functions are essential for non-trivial zero-knowledge. In *Proceedings of the 2nd Israel Symposium on Theory and Computing Systems*, ISTCS'93, pages 3–17, June 1993.
- 31 O. Reingold, L. Trevisan, and S. Vadhan. Notions of Reducibility between Cryptographic Primitives. In *Theory of Cryptography, First Theory of Cryptography Conference, TCC 2004, Cambridge, MA, USA, February 19-21, 2004, Proceedings*, pages 1–20, 2004.
- 32 J. Rompel. One-way Functions Are Necessary and Sufficient for Secure Signatures. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing*, pages 387–394, 1990.
- 33 R. Santhanam. Pseudorandomness and the Minimum Circuit Size Problem. In *11th Innovations in Theoretical Computer Science Conference, ITCS 2020*, volume 151 of *LIPIcs*, pages 68:1–68:26, 2020.
- 34 D. Xiao. *New Perspectives on the Complexity of Computational Learning, and Other Problems in Theoretical Computer Science*. PhD thesis, Princeton University, 2009.
- 35 D. Xiao. On basing  $ZK \neq BPP$  on the hardness of PAC learning. In *Proceedings of the 24th Conference on Computational Complexity, CCC'09*, pages 304–315, 2009.



# Spoofing Linear Cross-Entropy Benchmarking in Shallow Quantum Circuits

**Boaz Barak**

School of Engineering and Applied Sciences, Harvard University, Cambridge, MA, USA

<http://boazbarak.org>

b@boazbarak.org

**Chi-Ning Chou**

School of Engineering and Applied Sciences, Harvard University, Cambridge, MA, USA

<http://cnchou.github.io>

chiningchou@g.harvard.edu

**Xun Gao**

Department of Physics, Harvard University, Cambridge, MA, USA

xungao@g.harvard.edu

---

## Abstract

The *linear cross-entropy benchmark* (Linear XEB) has been used as a test for procedures simulating quantum circuits. Given a quantum circuit  $C$  with  $n$  inputs and outputs and purported simulator whose output is distributed according to a distribution  $p$  over  $\{0, 1\}^n$ , the linear XEB fidelity of the simulator is  $\mathcal{F}_C(p) = 2^n \mathbb{E}_{x \sim p} q_C(x) - 1$ , where  $q_C(x)$  is the probability that  $x$  is output from the distribution  $C|0^n\rangle$ . A trivial simulator (e.g., the uniform distribution) satisfies  $\mathcal{F}_C(p) = 0$ , while Google's noisy quantum simulation of a 53-qubit circuit  $C$  achieved a fidelity value of  $(2.24 \pm 0.21) \times 10^{-3}$  (Arute et. al., Nature'19).

In this work we give a classical randomized algorithm that for a given circuit  $C$  of depth  $d$  with Haar random 2-qubit gates achieves in expectation a fidelity value of  $\Omega(\frac{n}{L} \cdot 15^{-d})$  in running time  $\text{poly}(n, 2^L)$ . Here  $L$  is the size of the *light cone* of  $C$ : the maximum number of input bits that each output bit depends on. In particular, we obtain a polynomial-time algorithm that achieves large fidelity of  $\omega(1)$  for depth  $O(\sqrt{\log n})$  two-dimensional circuits. This is the first such result for two dimensional circuits of super-constant depth. Our results can be considered as an evidence that fooling the linear XEB test might be easier than achieving a full simulation of the quantum circuit.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Quantum complexity theory

**Keywords and phrases** Quantum supremacy, Linear cross-entropy benchmark

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.30

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/2005.02421>.

**Funding** *Boaz Barak*: Supported by NSF awards CCF 1565264 and CNS 1618026, a Simons Investigator Fellowship, and DARPA award W911NF2010021.

*Chi-Ning Chou*: Supported by NSF awards CCF 1565264 and CNS 1618026, a Simons Investigator Fellowship, and DARPA award W911NF2010021.

*Xun Gao*: Supported by the Postdoctoral Fellowship in Quantum Science of the Harvard-MPQ Center for Quantum Optics, the Templeton Religion Trust grant TRT 0159, and by the Army Research Office under Grant W911NF1910302 and MURI Grant W911NF-20-1-0082.

**Acknowledgements** We thank Scott Aaronson for helpful discussions.

## 1 Introduction

*Quantum computational supremacy* refers to experimental violations of the extended Church Turing Hypothesis using quantum computers. The most famous (and arguably at this point the only) example of such an experiment was carried out by Google [2]. The Google team



© Boaz Barak, Chi-Ning Chou, and Xun Gao;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 30; pp. 30:1–30:20



Leibniz International Proceedings in Informatics

LIPIcs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

constructed a device  $D$  that provides a “noisy simulation” of a quantum circuit  $C$  with  $n$  inputs and  $n$  outputs. The device can be thought as a “black box” that samples from a distribution  $p_D$  over  $\{0, 1\}^n$  that (loosely) approximates the distribution  $q_C$  that corresponds to measuring  $C$  applied to the all-zeroes string  $0^n$ . The quality of the device was measured using a certain benchmark known as the *Linear Cross-Entropy* benchmark (*a.k.a.* Linear XEB). The computational hardness assumption underlying the experiment is that no efficient classical algorithm can achieve a similar score. In this paper we investigate this assumption, giving a new classical algorithm for “spoofing” this benchmark in certain regimes. While our algorithm falls short of spoofing the benchmark in the parameter regime corresponding to the Google experiment, we do manage to achieve non-trivial results for deeper circuits than were known before. To our knowledge, this is the first algorithm that directly targets the linear XEB benchmark, without going through a full simulation of the underlying quantum circuit. Thus our work can be viewed as evidence that obtaining non-trivial performance for this benchmark is *not* equivalent to simulating quantum circuits.

The linear XEB benchmark is defined as follows. Let  $C$  be an  $n$ -qubit quantum circuit and  $q_C : \{0, 1\}^n \rightarrow [0, 1]$  be the pdf of the distribution obtained by measuring  $C|0^n\rangle$ . For each  $x \in \{0, 1\}^n$ , the instance linear XEB of  $x$  is defined as  $\mathcal{F}_C(x) := 2^n q_C(x) - 1$ . For every probability distribution  $p$ , the linear XEB fidelity of  $p$  with respect to circuit  $C$  is defined as

$$\mathcal{F}_C(p) := \mathbb{E}_{x \sim p} [\mathcal{F}_C(x)] = 2^n \sum_{x \in \{0, 1\}^n} q_C(x) p(x) - 1.$$

If  $C$  is a fully random circuit, then in expectation a perfect simulation  $p = q_C$  achieves  $\mathcal{F}_C(p) = 2$ .<sup>1</sup> Google’s “quantum computational supremacy” experiment demonstrated a noisy simulator sampling from a distribution  $p$  with  $\mathcal{F}_C(p) \approx (2.24 \pm 0.21) \times 10^{-3}$  for two dimensional 53-qubit circuits of depth 20. A trivial simulation (e.g. a distribution  $p$  which is the uniform distribution or another distribution independent of  $C$ ) will achieve  $\mathcal{F}_C(p) = 0$ . Motivated by the above, we say that  $p$  achieves *non-trivial fidelity* with respect to the circuit  $C$  if  $\mathcal{F}_C(p) = 1/\text{poly}(n)$ .<sup>2</sup>

The computational assumption underlying quantum computational supremacy with respect to some distribution  $\mathcal{D}$  over quantum circuits can be defined as follows. For every efficient randomized classical algorithm  $A$ , with high probability over  $C \sim \mathcal{D}$ , if we let  $A_C$  be the distribution of  $A$ ’s output on input  $C$ , then  $\mathcal{F}_C(A_C) = n^{-\omega(1)}$ . That is, the distribution output by  $A(C)$  has trivial fidelity with respect to  $C$ . Aaronson and Gunn [1] showed that this assumption follows from a (very strong) assumption they called “Linear Cross-Entropy Quantum Threshold Assumption” or XQUATH.<sup>3</sup>

In this work, we present an efficient classical algorithm  $A$  that satisfies  $\mathcal{F}_C(A_C) = \Omega(1)$  for quantum circuit  $C$  sampled from a distribution with Haar random 2-qubit gates with small *light cones* (see Definition 4).<sup>4</sup> Specifically, we prove the following theorem:

<sup>1</sup> This follows since  $q_C$  is the *Porter Thomas distribution*. However,  $q_C$  is *not* the maximizer of  $\mathcal{F}_C(p)$ : a distribution  $p$  that has all its mass on the mode  $x$  of the distribution  $q_C$  will achieve  $\mathcal{F}_C(p) \geq \Omega(n)$  for fully random circuits, and even higher values for shallower circuits as we’ll see below.

<sup>2</sup> As mentioned above, for an ideal simulation in random circuits the fidelity will be a constant. For noisy quantum circuits such as Google’s, the fidelity is roughly  $\exp(-\epsilon s)$  where  $\epsilon$  is the level of noise per gate and  $s = \Theta(d \cdot n)$  is the number of gates in the circuit.

<sup>3</sup> While [1] state their result for  $\Omega(1)$  fidelity, their proof shows that the XQUATH assumption implies that classical algorithms can not achieve  $1/\text{poly}(n)$  empirical fidelity with  $\text{poly}(n)$  samples.

<sup>4</sup> If  $C$  is a quantum circuit and  $i$  is an output bit of  $C$ , then the *light cone* of  $i$  is the set of all input bits  $j$  that are connected to  $i$  via a path in the circuit. For general circuits the light cone size can be exponential in the depth, but for one or two dimensional circuits, of the type used in quantum supremacy experiment, the light cone size is polynomial in the depth.

► **Theorem 1** (Linear XEB for circuits with small light cones). *Let  $n, d, L \in \mathbb{N}$  and let  $\mathcal{D}$  be a distribution over  $n$ -qubit quantum circuits with (i) light cone size at most  $L$ , (ii) depth at most  $d$ , and (iii) Haar random 2-qubit gates. Then, there exists a classical randomized algorithm  $A$  running in  $\text{poly}(n, 2^L)$  time such that*

$$\mathbb{E}_{C \sim \mathcal{D}} [\mathcal{F}_C(A_C)] \geq (1 + 15^{-d})^{\lfloor \frac{n}{L} \rfloor} - 1.$$

For constant dimensional circuits (such as the 2D quantum architecture used by Google), Theorem 1 yields the following corollary:

► **Corollary 2** (Constant dimensional circuits). *Let  $n \in \mathbb{N}$  and  $d = O(\log n)$ . Let  $c \in \mathbb{N}$  be a constant and  $\mathcal{D}$  be the distribution of  $n$ -qubit  $c$ -dimensional circuits of depth  $d$  with Haar random 2-qubit gates. There is a randomized algorithm  $A$  running in time  $2^{O(d^c)}$  such that*

$$\mathbb{E}_{C \sim \mathcal{D}} \mathcal{F}_C(A_C) = 1/\text{poly}(n).$$

**Proof.** A  $c$ -dimensional of depth  $d$  circuit has light-cone of size  $L = O(d^c) = n^{o(1)}$  for  $d = O(\log n)$ . Let  $d = \alpha \log n$ . By plugging in the parameters of Theorem 1, we see that (using  $\log_2 15 < 4$  and  $n/L \geq n^{1-o(1)}$ ) the expected value of the fidelity is at least

$$(1 + 2^{-4d})^{n^{1-o(1)}} - 1 \geq \Omega\left(\frac{n^{1-o(1)}}{n^{4\alpha}}\right).$$

The right hand side is at least  $1/\text{poly}(n)$  for every constant  $\alpha$  and in fact is at least  $\omega(1)$  for  $\alpha < 1/4$ . ◀

The bounds of Corollary 2 do *not* correspond to the Google experiment where the depth is roughly comparable to  $\sqrt{n}$ , rather than logarithmic. However, prior works in the literature were only able to achieve good linear XEB performance for circuits of *constant* depth (see Section 1.2). More importantly (in our view) is that our bounds show that it may be possible to achieve good linear XEB performance without achieving a full simulation.

## 1.1 From expectation to concentration

In actual experiments, one measures the *empirical* linear XEB, obtained by sampling  $x_1, \dots, x_T$  independently from the distribution  $p$  and computing  $\frac{1}{T} \sum_{i=1}^T 2^n q_C(x_i) - 1$ . Thus in our classical simulation we want to go beyond achieving large *expected* linear XEB benchmark, to show that our algorithm  $A$  actually achieves non-trivial empirical linear XEB with probability at least inverse polynomial over the choice of the circuit and with a number of samples  $T$  that is at most polynomial in  $n$ . These probability bounds are more challenging to prove, and at the moment our results are weaker than the optimal bounds one can hope for.

### Probability over circuits

For bounding the probability over *circuits* we show in Section 5.1, that in the setting of Theorem 1, for logarithmic depth circuits, we can obtain  $1/\text{poly}(n)$  fidelity with probability at least  $1/\text{poly}(n)$ . We also obtain more general tradeoffs between the fidelity, probability, and depth, see Corollary 12. We conjecture that random circuits from the distributions we consider exhibit much better concentration, and fact that the fidelity sharply concentrates around its expectation.

### Sample complexity, or probability over the algorithms' randomness

Bounding the *sample complexity* of our algorithm is a more difficult task than the expectation analysis because it requires higher moment information on  $\mathcal{F}_C(A_C)$ . We obtain only partial bounds in this setting, which we believe to be far from optimal. In Section 6 we show that an upper bound for the *collision probability* of  $q_C$  is sufficient to give an upper bound for the sample complexity of our algorithm. Specifically, letting  $CP(q) = \sum_{x \in \{0,1\}^n} q(x)^2$ , we show that if  $CP(q) = M \cdot 2^{-n}$  then the number of samples needed for the empirical linear XEB to achieve a value of at least  $\epsilon$  is  $M \cdot \exp(O(\epsilon \cdot 15^d))$ . In particular, for logarithmic depth circuits we can get inverse-polynomial empirical fidelity using  $O(M)$  samples. For *random* quantum circuits, where  $q_C$  is the Porter-Thomas distribution (with  $q_C(x)$  drawn independently as the square of a mean zero variance  $2^{-n}$  normal variable), it is known that  $CP(q_C) = O(2^{-n})$ , i.e.,  $M = O(1)$ . For shallow circuits, of the type we study, we show in Lemma 16 that  $CP(q_C) = O(2^{-n})$  for random *one dimensional* circuits of depth at least  $c \log n$  for some constant  $c > 0$ , which shows that we can achieve for such circuits inverse polynomial empirical fidelity using a polynomial number of samples. While this is significantly more technically challenging to prove, we conjecture that the same collision probability bound holds for *two dimensional* circuits of depth  $\Omega(\sqrt{\log n})$ . This conjecture, if true, will imply that for such circuits we can achieve  $1/\text{poly}(n)$  empirical fidelity using a polynomial number of samples, and constant fidelity using a sub-exponential (e.g.  $\exp\{\exp\{O(\sqrt{\log n})\}\}$ ) number of samples.<sup>5</sup>

## 1.2 Prior works

Prior classical algorithms mostly focused on the task of obtaining a full simulation (sampling from  $C|0^n\rangle$  or from a distribution close to it in statistical distance). We are not aware of any prior work that directly targeted the linear XEB measure and gave explicit bounds for the performance in this measure that are not implied by approximating the full distribution.

Napp et al [8] gave an algorithm to simulate random two-dimensional circuits of some small constant depth. They gave strong theoretical evidence that up to a certain constant depth, such circuits can be approximated by 1D circuits of small entanglement (i.e., “area law” as opposed to “volume law”), which can be effectively simulated using Matrix Product States [10]. However, [8] also gave evidence that the system undergoes a *phase transition* when the depth is more than some constant size (around 4), at which case the entanglement grows according to a “volume law” and hence their methods cannot be used to simulate circuits of super-constant depth.

Another direction of approximating large quantum circuits has considered the effect of *noise*. Some restricted classes of noisy quantum circuits were shown to be simulated by polynomial time classical algorithms in [4, 11] (in contrast to their noiseless variant [3]). This was also extended to more general random circuits by [6]. Very recent work has given numerical results suggesting that states generated by noisy quantum circuits could be approximated by Matrix Product States or Operators under the state fidelity measure [12, 9].<sup>6</sup> Low degree Fourier expansions yield other candidates for approximating such quantum states [6, 4].

<sup>5</sup> Very recently we have learned that Dalzell, Hunter-Jones, and Brandao (personal communication) refuted Conjecture 10 though we have not yet had a chance to verify their proof. The refutation of Conjecture 10 would only rule out a specific approach to upper bound the sample complexity. It is still possible that our algorithm achieves  $1/\text{poly}(n)$  empirical fidelity using a polynomial number of samples for *two dimensional* circuits of depth  $\Omega(\sqrt{\log n})$ .

<sup>6</sup> [12] briefly discusses the linear XEB measure as well, see Figure 7 there.



## 2 Preliminaries

In this section we introduce some of the notions we use for quantum circuits, and in particular distributions of random quantum circuits of fixed architecture, as well as tensor networks for analyzing quantum circuits. We include this here since some of this notation, and in particular tensor networks, might be unfamiliar to theoretical computer science audience. However, the reader can choose to skip this section and refer back to it as needed.

For  $n \in \mathbb{N}$ , an  $n$ -qubit quantum state  $|\psi\rangle = \sum_{x \in \{0,1\}^n} \alpha_x |x\rangle$  is a unit vector in  $\mathbb{C}^{2^n}$ . We let  $I, X, Y, Z$  denote the Pauli matrices where

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad Y = \begin{bmatrix} 0 & i \\ -i & 0 \end{bmatrix}, \quad \text{and} \quad Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}.$$

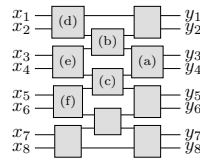
The following definition captures the notion of an “architecture” of a quantum circuit (see Figure 1 for an example):

► **Definition 3** (Circuit skeleton and light cone). *Let  $n \in 2\mathbb{N}$  and  $d \in \mathbb{N}$ , an  $n$ -qubit depth  $d$  circuit skeleton  $\mathcal{S}$  is a directed acyclic graph with  $d+2$  layers with the following structure. For convenience, we start the index of layers from 0.*

- *The  $0^{\text{th}}$  and the  $(d+1)^{\text{th}}$  layer has  $n$  nodes corresponding to the  $n$  input and output qubits. Each node in the first layer has exactly one out-going edge to the next layer while each node in the last layer has exactly one in-going edge from the previous layer.*
- *Each of the other layers has exactly  $n/2$  nodes and each node has exactly two in-going to the next layer and two out-going edges from the previous layer. Specifically, the first edge  $i^{\text{th}}$  gate is indexed by  $2i-1$  while the second edge is indexed by  $2i$  for each  $i = 1, 2, \dots, n/2$ .*
- *For each  $i = 1, 2, \dots, n$ , the  $i^{\text{th}}$  input node connects to the  $i^{\text{th}}$  edge of the second layer while the  $i^{\text{th}}$  edge of the  $(d+1)^{\text{th}}$  layer connects to the  $i^{\text{th}}$  output node.*

*Note that with the above definition, a circuit skeleton  $\mathcal{S}$  can be specified by  $d+1$  many permutations  $\pi^{(0)}, \pi^{(1)}, \dots, \pi^{(d)} \in S_n$ . Namely, for each  $t = 0, 1, 2, \dots, d-1$  and  $i = 1, 2, \dots, n$ , the  $i^{\text{th}}$  edge of the  $t^{\text{th}}$  layer connects to the  $\pi^{(t)}(i)^{\text{th}}$  edge of the  $(t+1)^{\text{th}}$  layer.*

*For every circuit skeleton  $G$ , the light cone size of  $G$  is the maximum over all output qubits  $i$  of the size of the set  $\{j : j \text{ is input qubit connected to } i \text{ in } G\}$ .*



■ **Figure 1** An example of 1D circuit skeleton with  $n = 8$  and  $d = 3$ . In this example the permutations are  $\pi^{(0)} = \pi^{(4)} = \text{id}$ ,  $\pi^{(1)} = \pi^{(3)} = (18765432)$ ,  $\pi^{(2)} = (81234567)$ .

Next, we define the light cone for an output qubit and the light cone size for a circuit skeleton.

► **Definition 4** (Light cone). *Let  $\mathcal{S}$  be a circuit skeleton and  $i$  be an output qubit. The light cone of  $i$  is the set of all input vertices in  $\mathcal{S}$  that has a path from left to right that ends at  $i$ . The light cone size of  $\mathcal{S}$  is then defined as the largest light cone size of an output qubit in  $\mathcal{S}$ .*

Note that the light cone size of the 1D circuit in Figure 1 is 6, which is less than the number of qubits. Also, it turns out that computing the marginal of an output qubit only requires the information from the light cone.

► **Lemma 5** (Marginal probability and light cone). *Let  $\mathcal{S}$  be a circuit skeleton with light cone size  $L$  and  $C$  be a circuit using skeleton  $\mathcal{S}$ . For each output qubit of  $C$ , the marginal probability can be computed in time  $O(2^L)$ .*

**Proof of Lemma 5.** We use the circuit skeleton in Figure 1 as an illustrating example. For an output qubit in  $C$ , to compute its marginal probability it suffices to compute the input state to the gate it connects to. For example, for output qubit  $y_3$ , it suffices to compute the input state to gate (a).

Similarly, to compute the input state of a gate, it suffices to compute the input states of the gate it connects to from the previous layer. Namely, to compute the input state of gate (a), it suffices to compute that of gate (b) and (c). If we continue this process inductively, the only input state needed to compute the marginal probability of an output bit is then the one lies in its light cone. In this example, to compute the marginal probability of  $y_3$ , it suffices to consider only  $x_1, x_2, \dots, x_6$ .

Finally, to compute the input states of all the intermediate gates, it suffices to perform  $2^L \times 2^L$  matrix vector multiplication because each intermediate state is of size at most  $L$ . While all the above operations can be done in  $O(2^L)$  times, computing the marginal probability of an output qubits in  $C$  only requires  $O(2^L)$  time. ◀

Now, we are able to formally define random quantum circuits.

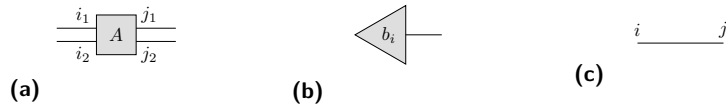
► **Definition 6** (Random quantum circuits). *Let  $n \in 2\mathbb{N}$ ,  $d \in \mathbb{N}$ . A distribution  $\mathcal{D}$  of  $n$ -qubit depth  $d$  random circuits consists of an  $n$ -qubit depth  $d$  circuit skeleton  $\mathcal{S}$  and ensembles  $\{\mathcal{E}_{i,j}\}$  over  $4 \times 4$  unitary matrices for each  $i = 1, \dots, d$  and  $j = 1, \dots, n/2$ .*

*A random quantum circuit  $C$  sampled from  $\mathcal{D}$  by sampling a  $4 \times 4$  unitary matrix  $U_i^{(t)}$  from  $\mathcal{E}_i^{(t)}$  and assigning  $U_i^{(t)}$  to the  $i^{\text{th}}$  node of the  $t^{\text{th}}$  layer for each  $t = 1, \dots, d$  and  $i = 1, \dots, n/2$ .*

*Specifically, if each  $\mathcal{E}_i^{(t)}$  is Haar random, then we say  $\mathcal{D}$  is Haar random 2-qubit circuits over  $\mathcal{S}$ .*

## 2.1 Tensor networks

Tensor network is an intuitive graphical language that can be rigorously used in reasoning about multilinear maps. Especially, it finds many applications in quantum computing since the basic operations such as partial measurement are all multilinear maps. In this paper, we restrict our attention to *qubits* (as opposed to the general case of *qudits*) and only to gates that act on two qubits.



■ **Figure 2** Three basic elements in tensor networks. (a) Gate: the figure represents  $\sum_{b_i, b_j \in \{0,1\}} A_{b_i, b_j} |b_i\rangle \langle b_j|$ . (b) State: the figure represents  $\langle b_i|$ . (c) Line: the figure represents  $\delta_{b_i, b_j}$ .

In a tensor network, we represent a unitary matrix (e.g., a gate) as a box with lines on the sides (see Figure 2a). Each line represents a coordinate of the gate and in this paper each coordinate has dimension 2 and is indexed by  $\{0, 1\}$ . Specifically, a line on the left represents a column vector (i.e.,  $|\cdot\rangle$ ) while a line on the right represents a row vector (i.e.,  $\langle \cdot|$ ).<sup>7</sup> For example, Figure 2a represents  $\sum_{b_i, b_j \in \{0,1\}} A_{b_i, b_j} |b_i\rangle \langle b_j|$ .

<sup>7</sup> This is when the tensor network is written left to right - sometimes it is written top to bottom, in which case a line on the top represents a column vector and a line on the bottom represents a row vector.

Similarly, a state (e.g., a qubit) is represented by a triangle with line only on one side and it is a  $|\cdot\rangle$  (resp.  $\langle\cdot|$ ) if the free-end of the line is left (resp. right). For example, Figure 2b represents  $\langle b_i|$ .

Semantically, a pure line refers to an indicator function<sup>8</sup> for its two ends. For example, the line in Figure 2c reads as  $\sum_{b_i, b_j \in \{0,1\}} \delta_{b_i, b_j} \langle b_i|b_j\rangle$  where  $\delta_{b_i, b_j} = 1$  if  $b_i = b_j$ ; otherwise it is 0.

### 3 Our Algorithm

We now describe our classical algorithm that spoofs the linear cross-entropy benchmark in shallow quantum circuits. The key idea is that rather than directly simulating the whole quantum circuit, our algorithm only computes the marginal distributions of few output qubits and then samples substrings for those qubit accordingly. We sample the remaining subits uniformly at random. Intuitively, due to the correlation on those output qubits, one can expect that the linear cross-entropy of our algorithm could be better than uniform distribution, but the analysis is somewhat delicate. Because consider shallow quantum circuits (of at most logarithmic light cone size), the marginal of few output qubits can be efficiently computed.

■ **Algorithm 1** Classical algorithm for spoofing linear XEB in shallow quantum circuits.

**Input:** A quantum circuit  $C$  sampled from  $\mathcal{D}_{\mathcal{S}}$ , a Haar random distribution over an  $n$ -qubit circuit skeleton  $\mathcal{S}$  with light cone size at most  $L$ .

- 1: We set  $m$  be some parameter in  $\{1, \dots, \lfloor n/L \rfloor\}$ . (We set  $m = \lfloor n/L \rfloor$  to obtain the result of Theorem 1 as stated.)
- 2: Find  $m$  output qubits  $i_1, \dots, i_m$  such that their light cones are disjoint.
- 3: Calculate the marginal probability of each output qubits  $i_1, \dots, i_m$ .
- 4: Sample  $x_{i_1}, \dots, x_{i_m}$  according to the marginal probabilities calculated in the previous step. For any  $i \notin \{i_1, \dots, i_m\}$ , sample  $x_i$  uniformly random from  $\{0, 1\}$ .

**Output:**  $x$ .

#### Running time of the algorithm

The total running time of Algorithm 1 is at most  $\text{poly}(n, 2^L)$ . Finding  $m$  outputs with disjoint light cones takes  $\text{poly}(n)$  time by a greedy algorithm. The second step takes  $\text{poly}(n, 2^L)$  time because it suffices to keep track of the  $2^L \times 2^L$  density matrix recording the marginal probability of every qubit in the light cone of  $i_j$  for each  $j \in [m]$  (see Lemma 5). The final step of sampling uniform bits for the remaining outputs can be done in polynomial time.

### 3.1 Analysis

The following theorem implies Theorem 1 by setting  $m = \lfloor n/L \rfloor$ :

► **Theorem 7** (Linear XEB for circuits with small light cones.). *Let  $n, d, L \in \mathbb{N}$  and let  $\mathcal{D}$  be a distribution over  $n$ -qubit quantum circuits with (i) light cone size at most  $L$ , (ii) depth at most  $d$ , and (iii) Haar random 2-qubit gates. Then, letting  $A_C$  be the distribution output by Algorithm 1 on input  $C$ ,*

$$\mathbb{E}_{C \sim \mathcal{D}} [\mathcal{F}_C(A_C)] \geq (1 + 15^{-d})^m - 1,$$

where  $m$  is the parameter chosen in step 1 of the algorithm.

<sup>8</sup> Also known as *contraction*.

The proof of Theorem 7 consists of three steps:

1. We reduce analyzing the expectation when the algorithm samples the marginals of  $m$  output qubits into analyzing it for a single output qubit.
2. We apply the integration formula for Haar measure and rewrite the expected linear XEB of a single qubit into a tensor network.
3. We then perform a change of basis on the tensor network and turn the single qubit analysis into a Markov chain problem where the expected linear XEB of a single qubit can be easily lower bounded.

Since the heart of the proof is the single output qubit analysis, we will describe it first.

#### 4 Single qubit analysis

In this section, we prove the  $m = 1$  case of our algorithm. That is, we prove that for a single output qubit, the expected contribution to linear XEB is of the order of  $15^{-d}$ .

► **Theorem 8** (Linear XEB of a single output qubit). *Let  $n, d \in \mathbb{N}$  and  $\mathcal{D}$  be distribution over  $n$ -qubit quantum circuits with depth at most  $d$  and with Haar random 2-qubit gates. For  $C \sim \mathcal{D}$ , let  $U$  denote the unitary matrix computed by  $C$ . For each  $i \in [n]$ , we have*

$$\mathbb{E}_{C \sim \mathcal{D}} [q_{C,i,0}^2 + q_{C,i,1}^2] \geq \frac{1 + 15^{-d}}{2},$$

where  $q_{C,i,b} = \Pr_{x \sim q_C} [x_i = b]$ .

We prove Theorem 8 by reducing to a Markov chain problem using tensor networks. Without loss of generality we can assume  $i = 1$ . Observe that

$$q_{C,1,0}^2 + q_{C,1,1}^2 = \frac{1 + \text{tr} (Z \otimes I^{\otimes n-1} U^\dagger |0^n\rangle \langle 0^n| U)^2}{2} \quad (1)$$

So our goal is to show that

$$\mathbb{E}_{C \sim \mathcal{D}} \left[ \text{tr} (Z \otimes I^{\otimes n-1} U^\dagger |0^n\rangle \langle 0^n| U)^2 \right] \geq 15^{-d}. \quad (2)$$

Let us start with rewriting the trace term of Equation 2 into an equivalent tensor network expression as follows.

$$\text{tr} (Z \otimes I^{\otimes n-1} U^\dagger |0^n\rangle \langle 0^n| U)^2 = \begin{array}{c} \begin{array}{c} \langle 0^n | \pi^{(0)} \end{array} \begin{array}{c} U_1 \end{array} \begin{array}{c} \pi^{(1)} \end{array} \begin{array}{c} U_2 \end{array} \cdots \begin{array}{c} U_d \end{array} \begin{array}{c} Z \otimes I^{\otimes n-1} \end{array} \\ \begin{array}{c} \langle 0^n | \pi^{(0)} \end{array} \begin{array}{c} U_1^\dagger \end{array} \begin{array}{c} \pi^{(1)} \end{array} \begin{array}{c} U_2^\dagger \end{array} \cdots \begin{array}{c} U_d^\dagger \end{array} \begin{array}{c} Z \otimes I^{\otimes n-1} \end{array} \\ \begin{array}{c} \langle 0^n | \pi^{(0)} \end{array} \begin{array}{c} U_1 \end{array} \begin{array}{c} \pi^{(1)} \end{array} \begin{array}{c} U_2 \end{array} \cdots \begin{array}{c} U_d \end{array} \begin{array}{c} Z \otimes I^{\otimes n-1} \end{array} \\ \begin{array}{c} \langle 0^n | \pi^{(0)} \end{array} \begin{array}{c} U_1^\dagger \end{array} \begin{array}{c} \pi^{(1)} \end{array} \begin{array}{c} U_2^\dagger \end{array} \cdots \begin{array}{c} U_d^\dagger \end{array} \begin{array}{c} Z \otimes I^{\otimes n-1} \end{array} \end{array}.$$

Next, for a single gate  $g$  in a quantum circuit, its expected behavior over the choice of 2-qubit Haar random gates can be characterized in the following lemma.

► **Lemma 9.** *Let  $U_g$  be Haar random 2-qubit gate, then the following holds.*

$$\mathbb{E}_{U_g} \left[ \begin{array}{c} \begin{array}{c} 1_a \\ 2_a \end{array} \begin{array}{c} U_g \end{array} \begin{array}{c} \tilde{1}_a \\ \tilde{2}_a \end{array} \\ \begin{array}{c} 1_b \\ 2_b \end{array} \begin{array}{c} U_g^\dagger \end{array} \begin{array}{c} \tilde{1}_b \\ \tilde{2}_b \end{array} \\ \begin{array}{c} 1_c \\ 2_c \end{array} \begin{array}{c} U_g \end{array} \begin{array}{c} \tilde{1}_c \\ \tilde{2}_c \end{array} \\ \begin{array}{c} 1_d \\ 2_d \end{array} \begin{array}{c} U_g^\dagger \end{array} \begin{array}{c} \tilde{1}_d \\ \tilde{2}_d \end{array} \end{array} \right] = \sum_{\sigma_1, \sigma_2, \sigma'_1, \sigma'_2 \in \{I, X, Y, Z\}} M_{\sigma_1, \sigma_2, \sigma'_1, \sigma'_2} \begin{array}{c} \begin{array}{c} 1_a \\ 1_b \\ 1_c \\ 1_d \end{array} \begin{array}{c} \sigma_1 \end{array} \begin{array}{c} \frac{1}{\sqrt{2}} \end{array} \\ \begin{array}{c} 2_a \\ 2_b \\ 2_c \\ 2_d \end{array} \begin{array}{c} \sigma_2 \end{array} \begin{array}{c} \frac{1}{\sqrt{2}} \end{array} \\ \begin{array}{c} \frac{1}{\sqrt{2}} \end{array} \begin{array}{c} \sigma'_1 \end{array} \begin{array}{c} \tilde{1}_a \\ \tilde{1}_b \\ \tilde{1}_c \\ \tilde{1}_d \end{array} \\ \begin{array}{c} \frac{1}{\sqrt{2}} \end{array} \begin{array}{c} \sigma'_2 \end{array} \begin{array}{c} \tilde{2}_a \\ \tilde{2}_b \\ \tilde{2}_c \\ \tilde{2}_d \end{array} \end{array}.$$

where

$$M_{\sigma_1, \sigma_2, \sigma'_1, \sigma'_2} = \begin{pmatrix} II & IX & IY & \cdots & ZZ \\ 1 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{15} & \frac{1}{15} & \cdots & \frac{1}{15} \\ 0 & \frac{1}{15} & \frac{1}{15} & \cdots & \frac{1}{15} \\ 0 & \vdots & \vdots & \ddots & \vdots \\ 0 & \frac{1}{15} & \frac{1}{15} & \cdots & \frac{1}{15} \end{pmatrix} \begin{matrix} II \\ IX \\ IY \\ \vdots \\ ZZ \end{matrix}.$$

The proof of Lemma 9 is based on the integration formula [5] for Haar measure. We postpone the proof of Lemma 9 to Subsection 4.1. Intuitively, the lemma says that by a change a basis, the expected behavior of a single Haar random 2-qubit gate can be exactly understood by an explicit transition matrix  $M$ . By the linearity of taking expectation, we can apply Lemma 9 on every gates in the circuit  $C$  and thus the whole tensor network is simplified to a *Markov chain*. Concretely, we have the following lemma.

► **Lemma 10** (Rewrite Equation 2 as a Markov chain). *Let  $n, d \in \mathbb{N}$  and  $\mathcal{D}$  be a Haar random distribution over an  $n$ -qubit depth  $d$  circuit skeleton  $\mathcal{S}$  with permutations  $\pi^{(0)}, \pi^{(1)}, \dots, \pi^{(d)}$ . For  $C \sim \mathcal{D}$ , let  $U$  denote the unitary matrix computed by  $C$ .*

$$\mathbb{E}_{C \sim \mathcal{D}} \left[ \text{tr} \left( Z \otimes I^{\otimes n-1} U^\dagger |0^n\rangle \langle 0^n| U \right)^2 \right] = \sum_{\substack{\sigma_{i'}^{(t')} \in \{I, X, Y, Z\} \\ i'=1,2,\dots,n \\ t'=1,2,\dots,d+1}} \prod_{t=0}^{d+1} V^{(t)} \left( \left\{ \sigma_{i'}^{(t')} \right\} \right) \quad (3)$$

where

$$V^{(t)} \left( \left\{ \sigma_{i'}^{(t')} \right\} \right) = \begin{cases} \prod_{i=1}^n \text{tr} \left( \frac{I+Z}{2} \frac{\sigma_i^{(1)}}{\sqrt{2}} \right)^2 & , \text{ if } t = 0 \\ \prod_{i=1}^{n/2} M_{\sigma_{2i-1}^{(t)}, \sigma_{2i}^{(t)}, \sigma_{\pi^{(t)}(2i-1)}^{(t+1)}, \sigma_{\pi^{(t)}(2i)}^{(t+1)}} & , \text{ if } t = 1, 2, \dots, d \\ \text{tr} \left( \frac{\sigma_1^{(d+1)}}{\sqrt{2}} Z \right)^2 \cdot \prod_{i=2}^n \text{tr} \left( \frac{\sigma_i^{(d+1)}}{\sqrt{2}} I \right)^2 & , \text{ if } t = d+1. \end{cases}$$

The proof of Lemma 10 is based on a careful composition of applying Lemma 9 on each of the gates. We postpone the proof of Lemma 10 to Subsection 4.2. Now, we are ready to prove Theorem 8 and complete the analysis for the expected linear XEB of single output qubit.

**Proof of Theorem 8.** Lemma 10 rewrites the desiring quantity into the form of a Markov chain so now it suffices to show that the right hand side of Equation 3 is at least  $15^{-d}$ .

Notice that for every possible assignment to  $\{\sigma_{i'}^{(t')}\}$ ,  $V(\{\sigma_{i'}^{(t')}\}) \geq 0$ . That is, it suffices to find an assignment such that  $\prod_{t=0}^{d+1} V^{(t)}(\{\sigma_{i'}^{(t')}\}) \geq 15^{-d}$ . Specifically, let us consider the following assignment. For all  $i = 1, 2, \dots, n$  and  $t = 1, 2, \dots, d+1$ , let

$$\sigma_i^{(t)} = \begin{cases} Z & , \text{ if } \pi^{(d)} \circ \pi^{(d-1)} \circ \dots \circ \pi^{(t)}(i) = 1 \\ I & , \text{ else.} \end{cases}$$

## 30:10 Spoofing Linear Cross-Entropy Benchmarking in Shallow Quantum Circuits

To analyze this assignment, let us start with the last layer. There we have  $\sigma_1^{(d+1)} = Z$  and  $\sigma_i^{(d+1)} = I$  for each  $i = 2, 3, \dots, n$  and thus

$$\begin{aligned} V^{(d+1)}(\{\sigma_{i'}^{(t')}\}) &= \text{tr} \left( \frac{\sigma_1^{(d+1)}}{\sqrt{2}} Z \right)^2 \cdot \prod_{i=2}^n \text{tr} \left( \frac{\sigma_i^{(d+1)}}{\sqrt{2}} I \right)^2 \\ &= \text{tr} \left( \frac{ZZ}{\sqrt{2}} \right)^2 \cdot \text{tr} \left( \frac{II}{\sqrt{2}} \right)^{2(n-1)} = 2^n. \end{aligned}$$

Next, for each  $t = 1, 2, \dots, d$  and  $i = 1, 2, \dots, n$ , observe that  $\sigma_i^{(t)} = \sigma_{\pi^{(t+1)}(i)}^{(t)}$  due to the choice of the assignment. As a result, all the  $M_{\sigma_{2i-1}^{(t)}, \sigma_{2i}^{(t)}, \sigma_{\pi^{(t)}(2i-1)}^{(t+1)}, \sigma_{\pi^{(t)}(2i)}^{(t+1)}}$  will be either  $M_{I,I,I,I}$  or  $M_{I,Z,I,Z}$ . Specifically, for each  $t = 1, 2, \dots, d$ , since there is exactly one  $Z$  appears among  $\{\sigma_i^{(t)}\}_{i=1,\dots,n}$  while the rest are  $I$ s, there is also exactly one  $M_{I,Z,I,Z}$  term contributes in  $V^{(t)}(\{\sigma_{i'}^{(t')}\})$  while the other terms are  $M_{I,I,I,I}$ . Namely, we have

$$V^{(t)}(\{\sigma_{i'}^{(t')}\}) = M_{I,Z,I,Z} \cdot (M_{I,I,I,I})^{n/2-1} = \frac{1}{15}$$

for each  $t = 1, 2, \dots, d$ .

Finally, since there is exactly one  $Z$  appears in  $\{\sigma_i^{(0)}\}_{i=1,\dots,n}$  while the rest are  $I$ s, we have

$$\begin{aligned} V^{(0)}(\{\sigma_{i'}^{(t')}\}) &= \prod_{i=1}^n \text{tr} \left( \frac{I + Z \sigma_i^{(0)}}{2 \sqrt{2}} \right)^2 = \text{tr} \left( \frac{I + Z}{2} \frac{Z}{\sqrt{2}} \right)^2 \cdot \text{tr} \left( \frac{I + Z}{2} \frac{I}{\sqrt{2}} \right)^{2(n-1)} \\ &= \left( \frac{1}{\sqrt{2}} \right) \cdot \left( \frac{1}{\sqrt{2}} \right)^{2(n-1)} = \frac{1}{2^n}. \end{aligned}$$

To sum up, we conclude that  $V(\{\sigma_{i'}^{(t')}\}) = \prod_{t=0}^{d+1} V^{(t)}(\{\sigma_{i'}^{(t')}\}) = 15^{-d}$  as desired. Specifically, this implies Equation 2, i.e.,  $\mathbb{E}_{C \sim \mathcal{D}} \left[ \text{tr} (Z \otimes I^{\otimes n-1} U^\dagger |0^n\rangle \langle 0^n| U)^2 \right] \geq 15^{-d}$ . Combine with Equation 1, this completes the proof of Theorem 8.  $\blacktriangleleft$

### 4.1 Proof of Lemma 9

We start with applying the integration formula for Haar random matrix and considering its tensor network representation.

► **Lemma 11** ([5, Equation 2.4]). *Let  $U$  be a Haar random 2-qubit gate, then we have the following. For each  $x_a, x_b, x_c, x_d, y_a, y_b, y_c, y_d \in \{0, 1\}^2$ ,*

$$\begin{aligned} \mathbb{E}_U [U_{x_a y_a} U_{x_b y_b}^\dagger U_{x_c y_c} U_{x_d y_d}^\dagger] &= \frac{1}{15} \cdot \left[ \delta_{x_a x_b} \delta_{x_c x_d} \delta_{y_a y_b} \delta_{y_c y_d} + \delta_{x_a x_d} \delta_{x_b x_c} \delta_{y_a y_d} \delta_{y_b y_c} \right] \\ &\quad - \frac{1}{60} \cdot \left[ \delta_{x_a x_b} \delta_{x_c x_d} \delta_{y_a y_d} \delta_{y_b y_c} + \delta_{x_a x_d} \delta_{x_b x_c} \delta_{y_a y_b} \delta_{y_c y_d} \right]. \end{aligned}$$

The above equation can be represented as the following tensor network.

$$\mathbb{E}_{U_g} \left[ \begin{array}{c} \text{1}_a \text{---} \boxed{U_g} \text{---} \widetilde{\text{1}}_a \\ \text{2}_a \text{---} \boxed{U_g} \text{---} \widetilde{\text{2}}_a \\ \text{1}_b \text{---} \boxed{U_g^\dagger} \text{---} \widetilde{\text{1}}_b \\ \text{2}_b \text{---} \boxed{U_g^\dagger} \text{---} \widetilde{\text{2}}_b \\ \text{1}_c \text{---} \boxed{U_g} \text{---} \widetilde{\text{1}}_c \\ \text{2}_c \text{---} \boxed{U_g} \text{---} \widetilde{\text{2}}_c \\ \text{1}_d \text{---} \boxed{U_g^\dagger} \text{---} \widetilde{\text{1}}_d \\ \text{2}_d \text{---} \boxed{U_g^\dagger} \text{---} \widetilde{\text{2}}_d \end{array} \right] = \frac{1}{15} \cdot \left[ \begin{array}{c} \text{1}_a \text{---} \text{---} \widetilde{\text{1}}_a \\ \text{1}_b \text{---} \text{---} \widetilde{\text{1}}_b \\ \text{1}_c \text{---} \text{---} \widetilde{\text{1}}_c \\ \text{1}_d \text{---} \text{---} \widetilde{\text{1}}_d \\ \text{2}_a \text{---} \text{---} \widetilde{\text{2}}_a \\ \text{2}_b \text{---} \text{---} \widetilde{\text{2}}_b \\ \text{2}_c \text{---} \text{---} \widetilde{\text{2}}_c \\ \text{2}_d \text{---} \text{---} \widetilde{\text{2}}_d \end{array} \right] - \frac{1}{60} \cdot \left[ \begin{array}{c} \text{1}_a \text{---} \text{---} \widetilde{\text{1}}_a \\ \text{1}_b \text{---} \text{---} \widetilde{\text{1}}_b \\ \text{1}_c \text{---} \text{---} \widetilde{\text{1}}_c \\ \text{1}_d \text{---} \text{---} \widetilde{\text{1}}_d \\ \text{2}_a \text{---} \text{---} \widetilde{\text{2}}_a \\ \text{2}_b \text{---} \text{---} \widetilde{\text{2}}_b \\ \text{2}_c \text{---} \text{---} \widetilde{\text{2}}_c \\ \text{2}_d \text{---} \text{---} \widetilde{\text{2}}_d \end{array} \right].$$

Next, the idea is to apply the Pauli identity (i.e., on each pair of  $(1_a, 1_b), (1_c, 1_d), (2_a, 2_b), (2_c, 2_d), (1'_a, 1'_b), (1'_c, 1'_d), (2'_a, 2'_b)$ , and  $(2'_c, 2'_d)$ ). Intuitively, this is doing a change of basis from the standard basis to Pauli basis.

Let us first apply the Pauli identity on  $(1_a, 1_b)$  and  $(1_c, 1_d)$  note that we have

$$\left( \begin{array}{c} 1a \\ 1b \\ 1c \\ 1d \end{array} \right) \left( \begin{array}{c} \sigma \\ \sigma' \end{array} \right) = \begin{cases} 2 \begin{array}{c} 1a \\ 1b \\ 1c \\ 1d \end{array} \left( \begin{array}{c} \sigma \\ \sigma' \end{array} \right), & \sigma = \sigma' \\ 0, & \text{else} \end{cases} \quad (4)$$

and

$$\left( \begin{array}{c} 1a \\ 1b \\ 1c \\ 1d \end{array} \right) \left( \begin{array}{c} \sigma \\ \sigma' \end{array} \right) = \begin{cases} 4 \begin{array}{c} 1a \\ 1b \\ 1c \\ 1d \end{array} \left( \begin{array}{c} \sigma \\ \sigma' \end{array} \right), & \sigma = \sigma' = I \\ 0, & \text{else.} \end{cases} \quad (5)$$

That is, the tensor network is non-zero only if  $\sigma = \sigma'$ . Thus, we only need one variable  $\sigma_1 \in \{I, X, Y, Z\}$  to handle  $(1_a, 1_b)$  and  $(1_c, 1_d)$ . Similarly, we can use  $\sigma_2, \widetilde{\sigma}_1, \widetilde{\sigma}_2 \in \{I, X, Y, Z\}$  to handle other pairs respectively. The equation becomes the following.

$$\mathbb{E}_{U_g} \left[ \begin{array}{c} 1a \\ 2a \\ 1b \\ 2b \\ 1c \\ 2c \\ 1d \\ 2d \end{array} \right] = \frac{1}{2^8} \sum_{\substack{\sigma'_1, \sigma'_2, \sigma''_1, \sigma''_2 \\ \in \{I, X, Y, Z\}}} \left( \frac{1}{15} \cdot \left[ \begin{array}{c} 1a \\ 1b \\ 1c \\ 1d \\ 2a \\ 2b \\ 2c \\ 2d \end{array} \right] + \left[ \begin{array}{c} 1a \\ 1b \\ 1c \\ 1d \\ 2a \\ 2b \\ 2c \\ 2d \end{array} \right] \right) - \frac{1}{60} \cdot \left[ \begin{array}{c} 1a \\ 1b \\ 1c \\ 1d \\ 2a \\ 2b \\ 2c \\ 2d \end{array} \right].$$

To finish the proof of Lemma 9, we have to explicitly calculate the value of the tensor network for each choice of  $\sigma_1, \sigma_2, \sigma_1', \sigma_2' \in \{I, X, Y, Z\}$ . Again, by Equation 4 and Equation 5, we have the following observations.

- If  $\sigma_1 = \sigma_2 = \sigma_1' = \sigma_2' = I$ , then the value is  $2^{-8} \cdot (15^{-1} \cdot (2^8 + 2^4) - 60^{-1} \cdot (2^6 + 2^6)) = 2^{-4}$ .
  - If  $\sigma_1 = \sigma_2 = I$  and at least one of  $\sigma_1', \sigma_2'$  is not  $I$ , or at least one of  $\sigma_1, \sigma_2$  is not  $I$  and  $\sigma_1' = \sigma_2' = I$ , then the value is  $2^{-4} \cdot (15^{-1} \cdot (0 + 2^4) + 60^{-1} \cdot (2^6 + 0)) = 0$ .
  - For all the other cases, the value is  $2^{-4} \cdot (15^{-1} \cdot (0 + 2^4) + 60^{-1} \cdot (0 + 0)) = 2^{-4} \cdot 15^{-1}$ .
- Finally, we take out the  $2^{-4}$  and evenly distribute it to the Pauli gates outside. Namely, each of them gets an extra  $1/\sqrt{2}$  factor as shown in the equation. This completes the proof of Lemma 9.

## 4.2 Proof of Lemma 10

Let us do a change of basis from the standard basis to the Pauli basis. Concretely, we apply Lemma 9 on every gate. Note that by the independence of each gate and the linearity of expectation, the  $t^{\text{th}}$  layer of the circuit becomes the following for each  $t = 1, 2, \dots, d$ .

$$\mathbb{E}_{U_t} \left( \begin{array}{c} 1a \\ n_a \\ 1b \\ n_b \\ 1c \\ n_c \\ 1d \\ n_d \end{array} \right) U_t = \sum_{\substack{\sigma_i^{(t)}, \widetilde{\sigma}_i^{(t)} \in \{I, X, Y, Z\} \\ \forall i=1, 2, \dots, n}} \prod_{i=1}^{n/2} M_{\sigma_{2i-1}^{(t)}, \sigma_{2i}^{(t)}, \widetilde{\sigma}_{2i-1}^{(t)}, \widetilde{\sigma}_{2i}^{(t)}} \left( \begin{array}{c} 1a \\ 1b \\ 1c \\ 1d \\ n_a \\ n_b \\ n_c \\ n_d \end{array} \right) \left( \begin{array}{c} \sigma_1^{(t)} \\ \sigma_2^{(t)} \\ \vdots \\ \sigma_n^{(t)} \end{array} \right) \left( \begin{array}{c} \widetilde{\sigma}_1^{(t)} \\ \widetilde{\sigma}_2^{(t)} \\ \vdots \\ \widetilde{\sigma}_n^{(t)} \end{array} \right).$$



### 30:12 Spoofing Linear Cross-Entropy Benchmarking in Shallow Quantum Circuits

Next, the  $i^{\text{th}}$  output wire at the  $t^{\text{th}}$  layer, i.e., the wires indexed by  $\tilde{i}_a, \tilde{i}_b, \tilde{i}_c, \tilde{i}_d$ , connects to the  $(\pi^{(t)}(i))^{\text{th}}$  input wire at the  $(t+1)^{\text{th}}$  layer, i.e., the wires indexed by  $\pi^{(t)}(i)_a, \pi^{(t)}(i)_b, \pi^{(t)}(i)_c, \pi^{(t)}(i)_d$ . By the orthogonality of Pauli gates (i.e., we have  $\text{tr}\left(\frac{\tilde{\sigma}_i^{(t)}}{\sqrt{2}} \frac{\sigma_{\pi^{(t)}(i)}^{(t+1)}}{\sqrt{2}}\right) = \delta_{\tilde{\sigma}_i^{(t)}, \sigma_{\pi^{(t)}(i)}^{(t+1)}}$  for all  $i = 1, 2, \dots, n$ . To sum up, the  $1^{\text{st}}$  to  $d^{\text{th}}$  layer is equivalent to following.

$$\begin{aligned}
 \mathbb{E}_{U_1, U_2, \dots, U_d} & \left[ \begin{array}{c} \begin{array}{cccc} \pi^{(0)} & \pi^{(1)} & \dots & \pi^{(d+1)} \\ \begin{array}{c} 1_a \\ n_a \\ \vdots \\ 1_b \\ n_b \\ \vdots \\ 1_c \\ n_c \\ \vdots \\ 1_d \\ n_d \end{array} & \begin{array}{c} U_1 \\ U_1^1 \\ \vdots \\ U_1^1 \\ U_1^1 \end{array} & \begin{array}{c} U_2 \\ U_2^1 \\ \vdots \\ U_2^1 \\ U_2^1 \end{array} & \begin{array}{c} U_d \\ U_d^1 \\ \vdots \\ U_d^1 \\ U_d^1 \end{array} \\ \end{array} \end{array} \right] = \\
 & \sum_{\substack{\sigma_i^{(t)} \in \{I, X, Y, Z\} \\ \forall i=1, 2, \dots, n \\ t=1, 2, \dots, t+1}} \prod_{t=1}^d \left( \prod_{i=1}^{n/2} M_{\sigma_{2i-1}^{(t)}, \sigma_{2i}^{(t)}, \sigma_{\pi^{(t)}(2i-1)}^{(t+1)}, \sigma_{\pi^{(t)}(2i)}^{(t+1)}} \right) \cdot \\
 & \begin{array}{c} \begin{array}{c} \sigma_1^{(1)} \\ \vdots \\ \sigma_n^{(1)} \end{array} \rightarrow \begin{array}{c} \sigma_1^{(d+1)} \\ \vdots \\ \sigma_n^{(d+1)} \end{array} \end{array}
 \end{aligned}$$

Finally, let us plug in the input and output layer. Recall that the input layer contains 4 copies of  $|0^n\rangle$  and the output layer contains 2 copies of  $Z \otimes I^{\otimes n-1}$ . Concretely, the contribution from the input layer would be

$$\prod_{i=1}^n \left( \langle 0 | \frac{\sigma_i^{(1)}}{\sqrt{2}} | 1 \rangle \right)^2 = \prod_{i=1}^n \text{tr} \left( \frac{I + Z}{2} \frac{\sigma_i^{(1)}}{\sqrt{2}} \right)^2$$

while the contribution from the output layer would be

$$\text{tr} \left( \frac{\sigma_1^{(d+1)}}{\sqrt{2}} Z \right)^2 \cdot \prod_{i=2}^n \text{tr} \left( \frac{\sigma_i^{(d+1)}}{\sqrt{2}} I \right)^2.$$

This completes the proof of Lemma 10.

## 5 Wrapping up: from single output bit to many bits

In this section we complete the proof of Theorem 1.

We will use the following notation. Let  $q(x)$  be a pdf over  $x \in \{0, 1\}^n$ . For any  $I \subset [n]$  and  $x_I \in \{0, 1\}^I$ , let  $q(I, x_I)$  denote the marginal probability of the output qubit at location  $I$  being  $x_I$ . Formally,  $q(I, x_I) = \sum_{y_{I^c} \in \{0, 1\}^{n-I}} q(y)$ . For a fixed input  $C$ , let  $I = \{i_1, \dots, i_m\}$  be the output qubits selected by Algorithm 1. Note that Algorithm 1 will choose the same  $I$  for each  $C$  sampled from  $\mathcal{D}$ . By the design of Algorithm 1,  $A_C(x) = \frac{1}{2^{n-m}} q_C(I, x_I)$  for every  $x \in \{0, 1\}^n$ . Thus, the linear XEB of  $A_C$  is the following.

$$\begin{aligned}
 \mathcal{F}_C(A_C) &= 2^n \sum_{x \in \{0, 1\}^n} q_C(x) A_C(x) - 1 = 2^n \sum_{x \in \{0, 1\}^n} q_C(x) \frac{q_C(I, x_I)}{2^{n-m}} - 1 \\
 &= 2^m \sum_{x_I \in \{0, 1\}^I} q_C(I, x_I)^2 - 1.
 \end{aligned} \tag{6}$$

Note that because their light cones are disjoint, we have  $q_C(I, x_I) = \prod_{j=1}^m q_C(\{i_j\}, x_{i_j})$ . Thus, the equation becomes

$$= \sum_{x_I \in \{0,1\}^I} \prod_{j=1}^m 2q_C(\{x_{i_j}\}, x_{i_j})^2 - 1. \quad (7)$$

Now, let us take expectation on the linear XEB over  $\mathcal{D}$ . Since  $\mathcal{D}$  fixes the structure of the circuit and the randomness only lies in the choice of gates,  $q_C(\{i_j\}, x_{i_j})$  is independent to each other. Namely,

$$\begin{aligned} \mathbb{E}_{C \sim \mathcal{D}} [\mathcal{F}_C(A_C)] &= \sum_{x_I \in \{0,1\}^I} \prod_{j=1}^m 2 \mathbb{E}_{C \sim \mathcal{D}} [q_C(\{x_{i_j}\}, x_{i_j})^2] - 1 \\ &= \prod_{j=1}^m 2 \mathbb{E}_{C \sim \mathcal{D}} [q_C(\{x_{i_j}\}, 0)^2 + q_C(\{x_{i_j}\}, 1)^2] - 1. \end{aligned} \quad (8)$$

Using the single qubit analysis (Theorem 8), we can complete the proof of Theorem 1 as follows.

**Proof of Theorem 1.** Apply Theorem 8 on Equation 8, we have

$$\mathbb{E}_{C \sim \mathcal{D}} [\mathcal{F}_C(A_C)] \geq (1 + 15^{-d})^m - 1$$

as desired. ◀

## 5.1 Probability over circuits

Using Theorem 1, we can obtain the following lower bound on the probability over the choice of the circuit  $C$  of obtaining non-trivial fidelity:

► **Corollary 12** (Lower bounding for the probability of success). *Let  $n, d, \mathcal{D}, L$  be as in Theorem 1. Then there is a randomized  $\text{poly}(n, 2^L)$  time algorithm such that for every  $1 \leq n \leq \lfloor \frac{n}{L} \rfloor$  and  $0 < \epsilon < 1$ ,*

$$\Pr_{C \sim \mathcal{D}} [\mathcal{F}_C(A_C) \geq ((1 + 15^{-d})^m - 1)] \geq \frac{(1 - \epsilon) \cdot ((1 + 15^{-d})^m - 1)}{2^m - 1} \geq \Omega\left(\frac{m \cdot 15^{-d}}{2^m}\right).$$

**Proof of Corollary 12.** The idea is simple - since our algorithm picks  $n - m$  bits uniformly at random, for every circuit  $C$ , by Equation 7,  $\mathcal{F}_C(A_C) \leq 2^m$ . Now, for any  $0 < \epsilon < 1$ , let  $\delta = \Pr_{C \sim \mathcal{D}} [\mathcal{F}_C(A_C) > \epsilon \cdot ((1 + 15^{-d})^m - 1)]$ , we have

$$(1 + 15^{-d})^m - 1 \leq \mathbb{E}_{C \sim \mathcal{D}} [\mathcal{F}_C(A_C)] \leq \delta \cdot 2^m + \epsilon \cdot ((1 + 15^{-d})^m - 1).$$

Thus,

$$\delta \geq \frac{(1 - \epsilon) \cdot ((1 + 15^{-d})^m - 1)}{2^m}. \quad \blacktriangleleft$$

## 6 Sample complexity analysis

In this section, we discuss the empirical linear XEB of our algorithm. Namely, how many samples are required so that the empirical average of the linear XEB can be non-trivially lower bounded. Specifically, the goal would be the following. For some  $T = \text{poly}(n)$ ,

$$\Pr_{\substack{C \sim \mathcal{D} \\ x_1, \dots, x_T \sim A_C}} \left[ \frac{1}{T} \sum_{i=1}^T \mathcal{F}_C(x_i) = \Omega(1) \right] \geq \frac{1}{\text{poly}(n)}. \quad (9)$$

In Section 5, we have shown that the expectation of the linear XEB of our algorithm is at least  $(1 + 15^{-d})^m$  for  $1 \leq m \leq \lfloor n/L \rfloor$  with probability  $1/\text{poly}(n)$  over the choice of random circuits. Thus, to achieve Equation 9, it suffices to show that the probability of the empirical average of the linear XEB deviating from  $\mathcal{F}_C(x)$  is small.

In general, it is a difficult task to rigorously upper bound the sample complexity of linear XEB. The reason is that such analysis needs to handle higher moment of  $q_C$  which is highly non-trivial for even 2D circuits. In this work we stick with the simpler case of analyzing the variance of linear XEB in Lemma 13. We further show in Lemma 14 that an inverse exponential bound on the collision probability of  $q_C$  would be sufficient for giving  $\text{poly}(n)$  upper bound for the sample complexity.

### 6.1 A variance/collision probability approach

The variance of  $\mathcal{F}_C(x)$  is sufficient for upper bounding the number of samples required for the empirical linear XEB to converge. Specifically, Chebyshev's inequality implies that with  $\text{Var}_{x \sim p}[\mathcal{F}_C(x)]/(\epsilon^2 \delta)$  many samples, the empirical XEB is at least  $\mathcal{F}_C(p) - \epsilon$  with probability  $\delta$  over the randomness of  $p$ . Note that here the circuit  $C$  is fixed.

► **Lemma 13.** *Let  $C$  be an  $n$ -qubit quantum circuit and  $q_C$  be the pdf of the distribution obtained from  $C |0^n\rangle$ . For any pdf  $p : \{0, 1\}^n \rightarrow [0, 1]$  and  $\epsilon, \delta \in (0, 1)$ , we have*

$$\Pr_{x_1, \dots, x_T \sim p} \left[ \frac{1}{T} \sum_{i=1}^T \mathcal{F}_C(x_i) \leq \mathcal{F}_C(p) - \epsilon \right] \leq \delta$$

when  $T \geq \frac{\text{Var}_{x \sim p}[\mathcal{F}_C(x)]}{\epsilon^2 \delta}$ .

**Proof.** Since  $\{\mathcal{F}_C(x_i)\}$  are i.i.d. random variables with mean  $\mathcal{F}_C(p)$  and variance  $\text{Var}_{x \sim p}[\mathcal{F}_C(x)]$ , by Chebyshev's inequality, we have

$$\Pr_{x_1, \dots, x_T \sim p} \left[ \frac{1}{T} \sum_{i=1}^T \mathcal{F}_C(x_i) < \mathcal{F}_C(p) - \epsilon \right] \leq \frac{\text{Var}_{x \sim p}[\mathcal{F}_C(x)]}{T \cdot \epsilon^2}.$$

As we pick  $T \geq \frac{\text{Var}_{x \sim p}[\mathcal{F}_C(x)]}{\epsilon^2 \delta}$ , the above error is at most  $\delta$  desired. ◀

The following lemma further shows that to upper bound the variance of our algorithm, it suffices to bound the *collision probability* of the ideal distribution.

► **Lemma 14.** *Let  $n, d, L \in \mathbb{N}$  and  $\mathcal{D}$  be distribution over  $n$ -qubit quantum circuits with (i) light cone size at most  $L$ , (ii) depth at most  $d$ , and (iii) with Haar random 2-qubit gates. Let  $1 \leq m \leq \lfloor n/L \rfloor$  and  $A$  be the algorithm from Algorithm 1, we have*

$$\text{Var}_{x \sim A_C} [\mathcal{F}_C(x)] \leq 2^{m+n} \sum_{x \in \{0, 1\}^n} q_C(x)^2$$

where  $\sum_{x \in \{0, 1\}^n} q_C(x)^2$  is also known as the collision probability of  $q_C$ .

**Proof of Lemma 14.** Consider the variance of the linear XEB of our algorithm  $A_C$  as follows.

$$\text{Var}_{x \sim A_C} [\mathcal{F}_C(x)] \leq \mathbb{E}_{x \sim A_C} [2^{2n} q_C(x)^2] = 2^{2n} \sum_{x \in \{0,1\}^n} A_C(x) q_C(x)^2.$$

Recall that  $A_C(x) \leq 2^{m-n}$  for all  $x$ , thus the equation becomes

$$\leq 2^{m+n} \sum_{x \in \{0,1\}^n} q_C(x)^2. \quad \blacktriangleleft$$

To have some intuition on Lemma 14, the right hand side is minimized when  $q_C$  is the uniform distribution over  $\{0,1\}^n$  where the collision probability is  $2^{-n}$ . In such case, the variance of our algorithm is  $O(2^m)$ . When choosing  $m = O(\log n)$ , the sample complexity of our algorithm would be  $\text{poly}(n)$  as desired.

In general, using the variance/collision probability to upper bound the sample complexity might not be tight. For example, consider the distribution of a sequence of independent biased coins, i.e.,  $q(x) = (1/2 + \epsilon)^{\|x\|_1} \cdot (1/2 - \epsilon)^{n - \|x\|_1}$  for each  $x \in \{0,1\}^n$ . Then the variance  $\text{Var}_{x \sim q}[q(x)]$  is exponentially large, however, the sample complexity of having the empirical average of  $q(x)$  being of the order of  $\mathbb{E}_{x \sim q}[q(x)]$  is  $O(1)$  with high probability. Specifically, when the depth of the random circuit is 1, then the marginal distribution looks like the above biased coins distribution with high probability and thus undesirable.

On the other extreme where the random circuit is very deep, it is known that the marginal distribution will converge to the *Porter Thomas distribution* and its collision probability is  $O(2^{-n})$  [2].

In Subsection 6.2, we further show that the collision probability of  $q_C$  is  $O(2^{-n})$  in expectation for 1D circuit of depth at least  $(\log n)/\log(5/4)$ . While the proof could potentially be extended to 2D circuit and beyond, we leave it as a future direction and state the following conjecture.

► **Conjecture 10.** Let  $n, d \in \mathbb{N}$  and  $\mathcal{D}$  be a distribution of  $n$ -qubit. For a circuit  $C$ , denote  $q_C$  as the pdf of  $C|0^n\rangle$ . We conjecture that there exists a constant  $c > 0$  such that when  $\mathcal{D}$  is the distribution over 2D random circuits of depth  $d \geq c\sqrt{\log n}$ ,

$$\mathbb{E}_{C \sim \mathcal{D}} \left[ \sum_{x \in \{0,1\}^n} q_C(x)^2 \right] = O\left(\frac{1}{2^n}\right).$$

## 6.2 The sample complexity of 1D random circuits of logarithmic depth

In this subsection, we formally prove that the sample complexity of our algorithm is  $O(2^m)$  for random 1D circuits with high probability.

► **Theorem 15.** Let  $n \in \mathbb{N}$  and  $\mathcal{D}$  be the distribution over  $n$ -qubit 1D quantum circuits with depth  $d = \Omega(\log n)$  and with Haar random 2-qudit gates where the dimension of the qudit is at least 4. Let  $1 \leq m \leq \lfloor n/2d \rfloor$  be the number of output qubits used by our algorithm. Then for any  $\delta \in (0, 1)$ , we have

$$\Pr_{C \sim \mathcal{D}} \left[ \text{Var}_{x \sim A_C} [\mathcal{F}_C(x)] = O\left(\frac{2^m}{\delta}\right) \right] \geq 1 - \delta.$$

Specifically, combine with Theorem 1, we have

$$\Pr_{\substack{C \sim \mathcal{D} \\ x_1, \dots, x_T \sim A_C}} \left[ \sum_{i=1}^T \mathcal{F}_C(x_i) = \Omega\left(\frac{1}{\text{poly}(n)}\right) \right] \geq \frac{1}{\text{poly}(n)}$$

when  $T = \Omega(1)$ .

### 30:16 Spoofing Linear Cross-Entropy Benchmarking in Shallow Quantum Circuits

**Proof of Theorem 15.** Let us first show that upper bounding the second moment of  $q_C$  is sufficient for proving Theorem 15. Consider the variance of the linear XEB of our algorithm  $A_C$  as follows.

$$\text{Var}_{x \sim A_C} [\mathcal{F}_C(x)] \leq \mathbb{E}_{x \sim A_C} [2^{2n} q_C(x)^2] = 2^{2n} \sum_{x \in \{0,1\}^n} A_C(x) q_C(x)^2.$$

Recall that  $A_C(x) \leq 2^{m-n}$  for all  $x$ , thus the equation becomes

$$\leq 2^{m+n} \sum_{x \in \{0,1\}^n} q_C(x)^2.$$

Next, the lemma below shows that the second moment term  $\sum_{x \in \{0,1\}^n} q_C(x)^2$  is exponentially small with high probability over the choice of  $C$ .

► **Lemma 16.** *Let  $n \in \mathbb{N}$  and  $\mathcal{D}$  be the distribution over  $n$ -qubit 1D quantum circuits with depth at least  $\frac{\log n}{\log(5/4)}$  and with Haar random 2-qubit gates. Then we have*

$$\mathbb{E}_{C \sim \mathcal{D}} \left[ \sum_{x \in \{0,1\}^n} q_C(x)^2 \right] = O\left(\frac{1}{2^n}\right).$$

The proof of Lemma 16 is based on the Ising model analysis by [7]. We postpone it to Subsection 6.3. Now, let us complete the proof of Theorem 15. By Lemma 16, we have

$$\mathbb{E}_{C \sim \mathcal{D}} \left[ \text{Var}_{x \sim A_C} [\mathcal{F}_C(x)] \right] \leq 2^{m+n} \mathbb{E}_{C \sim \mathcal{D}} [q_C(x)^2] \leq O(2^m).$$

Thus, for any  $\delta > 0$ , by Markov's inequality, we have  $\Pr_{C \sim \mathcal{D}} [\text{Var}_{x \sim A_C} [\mathcal{F}_C(x)] = O(2^m/\delta)] \geq 1 - \delta$  as desired. ◀

## 6.3 Proof of Lemma 16

It turns out that the previous Markov chain approach in analysis the expected linear XEB of our algorithm is not sufficient for upper bounding the expectation of  $\sum_{x \in \{0,1\}^n} q_C(x)^2$ . We thus consider a different approach by reducing the quantity to a combinatorial problem in a spin system on lattice. The proof is highly inspired by a recent paper of Hunter [7].

For the convenience of the analysis, here we fix the following skeleton for 1D circuit while the result can be easily extended to other variants.

$$(11)$$

### Step 1: Reducing to counting spin configurations on a hexagonal lattice

First, let us rewrite  $\sum_{x \in \{0,1\}^n} q_C(x)^2$  into an equivalent tensor network.

$$\begin{aligned} \sum_{x \in \{0,1\}^n} q_C(x)^2 &= \sum_{x \in \{0,1\}^n} \text{tr} (\langle 0^n | U | x \rangle \langle x | U^\dagger | 0^n \rangle)^2 \\ &= \sum_{x \in \{0,1\}^n} \begin{array}{c} \begin{array}{c} \leftarrow 0^n \pi^{(0)} U_1 \pi^{(1)} U_2 \cdots U_d \pi^{(d+1)} x \\ \leftarrow 0^n \pi^{(0)} U_1^\dagger \pi^{(1)} U_2^\dagger \cdots U_d^\dagger \pi^{(d+1)} x \\ \leftarrow 0^n \pi^{(0)} U_1 \pi^{(1)} U_2 \cdots U_d \pi^{(d+1)} x \\ \leftarrow 0^n \pi^{(0)} U_1^\dagger \pi^{(1)} U_2^\dagger \cdots U_d^\dagger \pi^{(d+1)} x \end{array} \end{array} \end{aligned}$$

In Lemma 9, we change the basis to the Pauli basis and replace the expectation of a single gate with a transition matrix. Here, we instead stick to the permutation basis in the integration formula and represent a single gate as an *effective vertex* [5].

► **Lemma 17** ([5]). *Let  $U_g$  be a Haar-random 2-qubit gate. We have the following.*

$$\mathbb{E}_{U_g} \left[ \begin{array}{c} 1_a \\ 2_a \\ 1_b \\ 2_b \\ 1_c \\ 2_c \\ 1_d \\ 2_d \end{array} \begin{array}{c} U_g \\ U_g^\dagger \\ U_g \\ U_g^\dagger \end{array} \begin{array}{c} \tilde{1}_a \\ \tilde{2}_a \\ \tilde{1}_b \\ \tilde{2}_b \\ \tilde{1}_c \\ \tilde{2}_c \\ \tilde{1}_d \\ \tilde{2}_d \end{array} \right] = \sum_{\sigma, \tau \in S_2} \begin{array}{c} 1_a, \dots, 1_d \quad 1'_a, \dots, 1'_d \\ \sigma \quad \tau \\ 2_a, \dots, 2_d \quad 2'_a, \dots, 2'_d \end{array}.$$

Here  $S_2 = \{\mathbb{I}, \mathbb{S}\}$  is the permutation group of two elements and an edge on the left represents four edges on the right. Specifically, the edge with  $\sigma$  and  $\tau$  on the two ends has weight  $\langle \sigma | \tau \rangle$  where

$$\langle \sigma | \tau \rangle = \begin{cases} \frac{1}{15} & , \sigma = \tau \\ \frac{-1}{60} & , \sigma \neq \tau. \end{cases}$$

for all  $\sigma, \tau \in S_2$ . As for the boundary condition, for each  $b_i \in \{0, 1\}$ , we have

$$\begin{array}{c} \begin{array}{c} b_i \\ b_i \\ b_i \\ b_i \\ b_i \\ b_i \\ b_i \\ b_i \end{array} \end{array} \begin{array}{c} \text{red circle} \\ \text{blue circle} \end{array} = \begin{array}{c} \text{red circle} \end{array}, \quad \begin{array}{c} \begin{array}{c} b_i \\ b_i \\ b_i \\ b_i \\ b_i \\ b_i \\ b_i \\ b_i \end{array} \end{array} \begin{array}{c} \text{blue circle} \\ \text{red circle} \end{array} = \begin{array}{c} \text{blue circle} \end{array}, \quad \text{and} \quad \sum_{\circ \in S_2} \begin{array}{c} \text{red circle} \end{array} \begin{array}{c} \text{blue circle} \end{array} = \frac{1}{20} \begin{array}{c} \text{red circle} \end{array}.$$

Apply Lemma 17 on a 1D circuit, the expectation of  $\sum_{x \in \{0,1\}^n} q_C(x)^2$  is then exactly the sum over spin configurations on the hexagonal lattice. For example, Equation 11 becomes the following. Note that each circle represents a distinct choices of elements from  $S_2$ .

$$\mathbb{E}_{C \sim \mathcal{D}} \left[ \sum_{x \in \{0,1\}^n} q_C(x)^2 \right] = 2^n \cdot \sum_{\circ, \circ \in S_2} \begin{array}{c} \begin{array}{c} \text{hexagonal lattice diagram} \end{array} \end{array}. \quad (12)$$

### Step 2: Reducing to counting domain wall configurations on a triangular lattice

The second idea in [7] is doing local summation on the blue vertices. Specifically, he showed that the local behavior of a blue vertex and its three red neighboring vertices can be fully described only by the spin of these three red vertices.

► **Lemma 18** ([7, Equation 18]). *Let  $U_g$  be a Haar-random 2-qubit gate. We have the following.*

$$\sum_{\tau \in S_2} \begin{array}{c} \text{red circle } \sigma_2 \\ \diagup \quad \diagdown \\ \text{blue circle } \tau \quad \text{red circle } \sigma_3 \\ \diagdown \quad \diagup \\ \text{red circle } \sigma_1 \end{array} = \begin{array}{c} \text{red circle } \sigma_2 \\ \diagup \quad \diagdown \\ \text{blue triangle} \\ \diagdown \quad \diagup \\ \text{red circle } \sigma_1 \end{array} = \begin{cases} 1 & , \sigma_1 = \sigma_2 = \sigma_3 \\ 0 & , \sigma_1 \neq \sigma_2 = \sigma_3 \\ \frac{2}{5} & , \text{else.} \end{cases}$$

An immediate corollary of Lemma 18 is that now we can instead summing over the spin configurations over a triangular lattice. That is, Equation 12 becomes the following

$$\mathbb{E}_{C \sim \mathcal{D}} \left[ \sum_{x \in \{0,1\}^n} q_C(x)^2 \right] = 2^n \cdot \left( \frac{1}{20} \right)^{n/2} \cdot \sum_{\sigma \in S_2} \begin{array}{c} \text{triangular lattice with red circles at vertices} \\ \text{and blue triangles at edges} \end{array} \quad (13)$$

The advantage of working on this triangular lattice is that the non-zero term on the right hand side of Equation 13 corresponds to a *domain wall* in the triangular lattice.

► **Definition 19** (Domain wall). *Consider the right hand side of Equation 13 and a configuration to all the red circles. The domain wall for this configuration is a collection of disjoint horizontal lines that separate the circles that are configured to  $\mathbb{I}$  from the circles that are configured to  $\mathbb{S}$ .*

Note that the domain wall configuration is in 1-to-1 correspondence with the spin configurations. Let  $\text{dw}$  denote a domain wall, let  $w(\text{dw})$  be the weight of the corresponding spin configuration. Thus, Equation 13 becomes the following.

$$\mathbb{E}_{C \sim \mathcal{D}} \left[ \sum_{x \in \{0,1\}^n} q_C(x)^2 \right] = \left( \frac{1}{5} \right)^{n/2} \cdot \sum_{\text{dw} \in \begin{array}{c} \text{triangular lattice} \\ \text{with domain walls} \end{array}} w(\text{dw}). \quad (14)$$

Note that a domain wall could contain two types of paths: (i) a path that goes from left boundary to the right boundary and (ii) a path that starts from and ends at both the right boundary. Furthermore, as the domain wall configuration is in 1-to-1 correspondence with subset of disjoint paths, Equation 14 becomes the following.

$$\mathbb{E}_{C \sim \mathcal{D}} \left[ \sum_{x \in \{0,1\}^n} q_C(x)^2 \right] \leq \left( \frac{1}{5} \right)^{n/2} \cdot \left( 1 + \sum_{\substack{\text{disjoint path of type (i)} \\ \in \begin{array}{c} \text{triangular lattice} \\ \text{with domain walls} \end{array}}} w(\text{path}) \right) \cdot \left( 1 + \sum_{\substack{\text{disjoint path of type (ii)} \\ \in \begin{array}{c} \text{triangular lattice} \\ \text{with domain walls} \end{array}}} w(\text{path}) \right). \quad (15)$$



### Step 3: Upper bound the sum of possible path configurations of type (i)

For a set of disjoint path of type (i), it contains at most  $\lfloor n/2 \rfloor$  paths. Also, a path of type (i) contributes  $\left(\frac{q}{q^2+1}\right)^d$  in the weight of a domain wall.

Now, for each  $1 \leq t \leq \lfloor n/2 \rfloor$  let us first estimate the number of domain walls having at most  $t$  paths of type (i). Specifically, for every  $i \in [\lfloor n/2 \rfloor]$ , the number of paths starting from  $i$  is at most  $2^d$  because at each layer it either moves up or down. Next, for each  $1 \leq t \leq \lfloor n/2 \rfloor$ , the number of possible  $t$  paths is then at most  $\binom{n/2}{t} \cdot 2^{dt}$ . This gives the following upper bound for the weight contributing from domain wall of type (i).

$$\begin{aligned} \left( \sum_{\substack{\text{disjoint path of type (i)} \\ \in \text{domain wall}}} w(\text{path}) \right) &\leq \sum_{t=1}^{\lfloor n/2 \rfloor} \left(\frac{2}{5}\right)^{dt} \cdot \binom{\lfloor n/2 \rfloor}{t} \cdot 2^{dt} \\ &\leq \sum_{t=1}^{\lfloor n/2 \rfloor} \left(\frac{4}{5}\right)^{dt} \cdot \binom{\lfloor n/2 \rfloor}{t} \\ &\leq \left(1 + \left(\frac{4}{5}\right)^d\right)^{\lfloor n/2 \rfloor}. \end{aligned}$$

Consider  $d \geq \frac{\log n}{\log(5/4)}$ , the equation becomes

$$\leq \exp\left(\left(\frac{4}{5}\right)^{\frac{\log n}{\log(5/4)}} \frac{n}{2}\right) = O(1).$$

### Step 4: Upper bound the sum of possible path configurations of type (ii)

Let us consider the 1D circuit with infinite depth and denote the distribution as  $\mathcal{D}_\infty$ . Also, since the depth is infinity, the sum of the weight of domain wall with paths of type (i) is negligible. Namely, only paths of type (ii) contribute in the infinite depth circuit. Thus, we have the following upper bound.

$$\begin{aligned} \left( 1 + \sum_{\substack{\text{disjoint path of type (ii)} \\ \in \text{domain wall}}} w(\text{path}) \right) &\leq \left( 1 + \sum_{\substack{\text{disjoint path of type (ii)} \\ \in \text{domain wall}}} w(\text{path}) \right) \\ &= 5^{n/2} \cdot \mathbb{E}_{C \sim \mathcal{D}_\infty} \left[ \sum_{x \in \{0,1\}^n} q_C(x)^2 \right]. \end{aligned}$$

Finally, it is a well known fact that the expectation of the sum of squares of marginal probabilities is  $O(2^{-n})$  for infinite depth 1D circuit. So the above equation becomes

$$= O\left(\left(\frac{5}{4}\right)^{n/2}\right).$$

**Wrap up**

To conclude the proof of Lemma 16, let us plug in the calculations from step 3 and step 4 into Equation 15, this gives us

$$\mathbb{E}_{C \sim \mathcal{D}} \left[ \sum_{x \in \{0,1\}^n} q_C(x)^2 \right] \leq \left( \frac{1}{5} \right)^{n/2} \cdot O(1) \cdot O \left( \left( \frac{5}{4} \right)^{n/2} \right) = O \left( \frac{1}{2^n} \right)$$

as desired.

---

**References**


---

- 1 Scott Aaronson and Sam Gunn. On the classical hardness of spoofing linear cross-entropy benchmarking. *arXiv preprint*, 2019. [arXiv:1910.12085](#).
- 2 Frank Arute, Kunal Arya, Ryan Babbush, Dave Bacon, Joseph C Bardin, Rami Barends, Rupak Biswas, Sergio Boixo, Fernando GSL Brandao, David A Buell, et al. Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779):505–510, 2019.
- 3 Michael J Bremner, Ashley Montanaro, and Dan J Shepherd. Average-case complexity versus approximate simulation of commuting quantum computations. *Physical review letters*, 117(8):080501, 2016.
- 4 Michael J Bremner, Ashley Montanaro, and Dan J Shepherd. Achieving quantum supremacy with sparse and noisy commuting quantum computations. *Quantum*, 1:8, 2017.
- 5 PW Brouwer and CWJ Beenakker. Diagrammatic method of integration over the unitary group, with applications to quantum transport in mesoscopic systems. *Journal of Mathematical Physics*, 37(10):4904–4934, 1996.
- 6 Xun Gao and Luming Duan. Efficient classical simulation of noisy quantum computation. *arXiv preprint*, 2018. [arXiv:1810.03176](#).
- 7 Nicholas Hunter-Jones. Unitary designs from statistical mechanics in random quantum circuits. *arXiv preprint*, 2019. [arXiv:1905.12053](#).
- 8 John Napp, Rolando L La Placa, Alexander M Dalzell, Fernando GSL Brandao, and Aram W Harrow. Efficient classical simulation of random shallow 2d quantum circuits. *arXiv preprint*, 2019. [arXiv:2001.00021](#).
- 9 Kyungjoo Noh, Liang Jiang, and Bill Fefferman. Efficient classical simulation of noisy random quantum circuits in one dimension. *arXiv preprint*, 2020. [arXiv:2003.13163](#).
- 10 Guifr  Vidal. Efficient simulation of one-dimensional quantum many-body systems. *Physical review letters*, 93(4):040502, 2004.
- 11 Man-Hong Yung and Xun Gao. Can chaotic quantum circuits maintain quantum supremacy under noise? *arXiv preprint*, 2017. [arXiv:1706.08913](#).
- 12 Yiqing Zhou, E Miles Stoudenmire, and Xavier Waintal. What limits the simulation of quantum computers? *arXiv preprint*, 2020. [arXiv:2002.07730](#).

# On the Complexity of Isomorphism Problems for Tensors, Groups, and Polynomials I: Tensor Isomorphism-Completeness

Joshua A. Grochow 

Departments of Computer Science and Mathematics, University of Colorado, Boulder, CO, USA  
jgrochow@colorado.edu

Youming Qiao

Centre for Quantum Software and Information, University of Technology Sydney, Australia  
youming.qiao@uts.edu.au

---

## Abstract

We study the complexity of isomorphism problems for tensors, groups, and polynomials. These problems have been studied in multivariate cryptography, machine learning, quantum information, and computational group theory. We show that these problems are all polynomial-time equivalent, creating bridges between problems traditionally studied in myriad research areas. This prompts us to define the complexity class  $TI$ , namely problems that reduce to the Tensor Isomorphism ( $TI$ ) problem in polynomial time. Our main technical result is a polynomial-time reduction from  $d$ -tensor isomorphism to 3-tensor isomorphism. In the context of quantum information, this result gives multipartite-to-tripartite entanglement transformation procedure, that preserves equivalence under stochastic local operations and classical communication (SLOCC).

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Complexity classes; Computing methodologies  $\rightarrow$  Linear algebra algorithms; Hardware  $\rightarrow$  Quantum communication and cryptography

**Keywords and phrases** complexity class, tensor isomorphism, polynomial isomorphism, group isomorphism, stochastic local operations and classical communication

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.31

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/1907.00309>.

**Funding** *Joshua A. Grochow*: Partially supported by NSF grants DMS-1750319 and DMS-1622390. *Youming Qiao*: Partially supported by the Australian Research Council DP200100950.

**Acknowledgements** We would like to thanks James B. Wilson for related discussions, and Uriya First, Lek-Heng Lim, and J. M. Landsberg for help searching for references asking whether  $dTI$  could reduce to  $3TI$ . We also thank Nengkun Yu, Yinan Li, and Graeme Smith for explaining the notion of SLOCC, and Ryan Williams for pointing out the problem of distinguishing between  $ETH$  and  $\#ETH$ .

## 1 Introduction

Although GRAPH ISOMORPHISM ( $GI$ ) is perhaps the most well-studied isomorphism problem in computational complexity – even going back to Cook’s and Levin’s initial investigations into NP (see [3, Sec. 1]) – it has long been considered to be solvable in practice [51, 52], and Babai’s recent quasi-polynomial-time breakthrough is one of the theoretical gems of the last several decades [5].

However, several isomorphism problems for tensors, groups, and polynomials seem to be much harder to solve, both in practice – they’ve been suggested as difficult enough to support cryptography [39, 57] – and in theory: the best known worst-case upper bounds are barely improved from brute force (e.g., [46, 63]). As these problems arise in a variety of areas, from



© Joshua A. Grochow and Youming Qiao;  
licensed under Creative Commons License CC-BY  
12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 31; pp. 31:1–31:19



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

multivariate cryptography and machine learning, to quantum information and computational algebra, getting a better understanding of their complexity is an important goal with many potential applications. These isomorphism problems are the focus of this paper.

Our first set of results shows that all these isomorphism problems from many research areas are equivalent under polynomial-time reductions, creating bridges between different disciplines. The TENSOR ISOMORPHISM (TI) problem turns out to occupy a central position among these problems, leading us to define the complexity class TI, consisting of those problems polynomial-time reducible to the TENSOR ISOMORPHISM problem.

More specifically, we first present a polynomial-time reduction from  $d$ -TENSOR ISOMORPHISM to 3-TENSOR ISOMORPHISM. This result may be viewed as corresponding to the  $k$ -SAT to 3-SAT reduction in the setting of TENSOR ISOMORPHISM, but the proof is much more involved. This result also has a natural application to quantum information: it gives a procedure that turns multipartite entanglements to tripartite entanglements while preserving equivalence under stochastic local operations and classical communication (SLOCC).

We then demonstrate that various isomorphism problems for polynomials, general algebras, groups, and tensors all turn out to be TI-complete. One important reference here is the recent work [26], in which they showed that several such problems reduce to 3TI. Our contribution is to show that these problems are also 3TI-hard. Another set of related works are [1, 2, 42] by Agrawal, Kayal, and Saxena, who showed some equivalences and reductions between RING ISOMORPHISM (commutative with unit), CUBIC FORM EQUIVALENCE, and isomorphism of commutative, unital, associative algebras [1, 2, 42]. Here we greatly expand these and show a much wider class of problems are equivalent (see Thm. 4=Thm. B and Fig. 1).

In a follow-up paper [33], we study search and counting to decision reductions, apply these results to GROUP ISOMORPHISM in the matrix group model, and obtain a nilpotency class reduction for GROUP ISOMORPHISM.

All these results together lay the foundation for an emerging theory of the complexity class TI that in some cases parallels, and in some cases deviates from, the complexity theory of the class GI, namely the set of problems that are polynomial-time reducible to GRAPH ISOMORPHISM [43]. From the theory perspective, this theory reveals a family of algorithmic problems demonstrating highly interesting complexity-theoretic properties. From the practical perspective, this theory could serve as guidance for, and facilitate dialogue among, researchers from diverse research areas including cryptography, machine learning, quantum information, and computational algebra. Indeed, some of our results already have natural applications to quantum information and computational group theory.

**Organization.** Due to page constraints and the nature of this work, we are only able to present the main results and the related implications and discussions. For detailed proofs, we refer the reader to the full version [32]. In the remainder of this paper, we first present the origins of those isomorphism problems we consider (Sec. 2). We then state our main results in Sec. 3, and briefly indicate the main techniques in Sec. 4. In Sec. 5, we present formal statements of the various problems involved and a detailed statement of one main result. Finally in Sec. 6 we present the implication to quantum information and discuss on some further related works and the outlook of this research direction.

## 2 Isomorphism testing problems from several areas

Let  $\mathbb{F}$  be a field. Let  $\text{GL}(n, \mathbb{F})$  denote the general linear group of degree  $n$  over  $\mathbb{F}$ , and  $\text{M}(n, \mathbb{F})$  the linear space of  $n \times n$  matrices. For a finite field  $\mathbb{F}_q$ , we may also write  $\text{GL}(n, \mathbb{F}_q)$  and  $\text{M}(n, \mathbb{F}_q)$  as  $\text{GL}(n, q)$  and  $\text{M}(n, q)$ , respectively.

**Multivariate cryptography.** In 1996, Patarin [57] proposed identification and signature schemes based on a family of problems called “isomorphism of polynomials.” A specific problem, called *isomorphism of (quadratic) polynomials with two secrets* (IP2S), asks the following. Let  $\vec{f} = (f_1, \dots, f_m)$  and  $\vec{g} = (g_1, \dots, g_m)$  be two  $m$ -tuples of homogeneous quadratic polynomials, where  $f_i, g_j \in \mathbb{F}[x_1, \dots, x_n]$ . Recall an  $m$ -tuple of polynomials in  $n$  variables can be viewed as a polynomial map from  $\mathbb{F}^n$  to  $\mathbb{F}^m$ . It is natural to ask whether  $\vec{f}$  and  $\vec{g}$  represent the same polynomial map up to change of basis, or more specifically, whether there exists  $P \in \text{GL}(n, \mathbb{F})$  and  $Q \in \text{GL}(m, \mathbb{F})$ , such that  $Q \circ \vec{f} \circ P = \vec{g}$ . Since then, the IP2S problem, and its variant isomorphism of (quadratic) polynomials with one secret (IP1S), have been intensively studied in multivariate cryptography (see [11, 38] and references therein).

**Machine learning.** In machine learning, it is natural to view a sequential data stream as a path. This leads to the use of the *signature* tensor of a path  $\phi : [0, 1] \rightarrow \mathbb{R}^n$ , first introduced by Chen [20] to extract features of data. This is the basic idea of the signature tensor method, which has been pursued by in a series of works; see [21, 49, 54] and references therein. The algorithmic problem of reconstructing the path from the signature tensor is of considerable interest; see e.g. [50, 59]. In this context, the following problem was recently studied by Pfeffer, Seigal, and Sturmfels [59], called the TENSOR CONGRUENCE problem: given two 3-tensors  $\mathbf{A} = (a_{ijk}), \mathbf{B} = (b_{ijk}) \in \mathbb{F}^{n \times n \times n}$ , decide whether there exists  $P \in \text{GL}(n, \mathbb{F})$ , such that the congruence action of  $P$  sends  $\mathbf{A}$  to  $\mathbf{B}$ . More specifically, this action of  $P = (p_{ij})$  sends  $\mathbf{A} = (a_{ijk})$  to  $\mathbf{A}' = (a'_{ijk})$ , where  $a'_{ijk} = \sum_{i', j', k'} a_{i'j'k'} p_{i,i'} p_{j,j'} p_{k,k'}$ .

**Quantum information.** Let  $\mathcal{H} = \mathcal{H}_1 \otimes \dots \otimes \mathcal{H}_d$ , where  $\mathcal{H}_i = \mathbb{C}^{n_i}$ . Let  $\rho = |\phi\rangle\langle\phi|$  and  $\tau = |\psi\rangle\langle\psi|$  be two pure quantum states, where  $|\phi\rangle, |\psi\rangle \in \mathcal{H}$ . In quantum information, a natural question is to decide whether  $\rho$  can be converted to  $\tau$  using local operations and classical communication statistically (SLOCC), i.e. with non-zero probability [10, 23]. It is well-known by [23] that  $\rho$  and  $\tau$  are interconvertible via SLOCC, if and only if there exist  $T_i \in \text{GL}(\mathcal{H}_i)$ , such that  $(T_1 \otimes \dots \otimes T_m)|\phi\rangle = |\psi\rangle$ . Therefore, given pure quantum states  $\rho$  and  $\tau$ , whether  $\rho$  and  $\tau$  are interconvertible via SLOCC can be cast as an isomorphism testing problem, called the  $d$ -TENSOR ISOMORPHISM problem (see Definition 1).

**Computational group theory.** In computational group theory, a notoriously difficult problem is to test isomorphism of finite  $p$ -groups, namely groups of prime power order (see, e.g., [55]). Here, the groups are represented succinctly, e.g., by generating sets of permutations or matrices over finite fields. Testing isomorphism of  $p$ -groups is considered to be a bottleneck to testing isomorphism of general groups [7, 19, 31]. Even for  $p$ -groups of class 2 and exponent  $p$ , current methods are still limited to instances of quite small size.

**Theoretical computer science.** As already mentioned, Agrawal, Kayal, and Saxena studied isomorphism and automorphism problems of rings, algebras, and polynomials [1, 2, 42], motivated by several problems including PRIMALITY TESTING, POLYNOMIAL FACTORIZATION, and GRAPH ISOMORPHISM. Later, motivated by cryptographic applications and algebraic complexity, Kayal studied the POLYNOMIAL EQUIVALENCE problems (possibly under affine projections) and solved certain important special cases [40, 41] (see also [30]). Among these problems, we will be mostly concerned with the following two. First, the ALGEBRA ISOMORPHISM problem for commutative, unital, associative algebras over a field  $\mathbb{F}$ , asks whether two such algebras, given by structure constants, are isomorphic. Second, the CUBIC FORM EQUIVALENCE problem asks whether two homogeneous cubic polynomials over  $\mathbb{F}$  are equivalent under the natural action of the general linear group by change of basis on the variables.

**Practical complexity of these problems.** The preceding isomorphism testing problems are of great interest to researchers from seemingly unrelated areas. Furthermore, they pose considerable challenges for practical computations at the present stage. The latter is in sharp contrast to GRAPH ISOMORPHISM, for which very effective practical algorithms have existed for some time [51, 52]. Indeed, the problems we consider have been proposed to be difficult enough for cryptographic purposes [39, 57]. As further evidence of their practical difficulty, current algorithms implemented for testing isomorphism of  $p$ -groups of class 2 and exponent  $p$  can handle groups of dimension 20 over  $\mathbb{F}_{13}$ , but absolutely not for groups of dimension 200 over  $\mathbb{F}_{13}$ , even though in this case the input can still be stored in only a few megabytes.<sup>1</sup> In [59, arXiv version 1], computations on special cases of the TENSOR CONGRUENCE problem were performed in Macaulay2 [28], but these could not go beyond small examples either.

**A note on terminology.** Before introducing our results formally, a terminological note is in order: we shall call valence- $d$  tensors  $d$ -way arrays, and tensors will be understood to be  $d$ -way arrays considered under a specific group action. The reason for this change of terminology will be clearer in the following. We remark that it is not uncommon to see such differences in the terminologies around tensors, see, e.g., the preface of [45].

We follow a natural convention: when  $\mathbb{F}$  is finite, a fixed algebraic extension of a finite field such as  $\overline{\mathbb{F}}_p$ , the rationals, or a fixed algebraic extension of the rationals such as  $\overline{\mathbb{Q}}$ , we consider the usual model of Turing machines; when  $\mathbb{F}$  is  $\mathbb{R}$ ,  $\mathbb{C}$ , the  $p$ -adic rationals  $\mathbb{Q}_p$ , or other more “exotic” fields, we work in the Blum–Shub–Smale model over  $\mathbb{F}$ .

### 3 Main results

#### 3.1 Defining the TENSOR ISOMORPHISM complexity class

Given the diversity of the isomorphism problems from Sec. 2, the first main question addressed in this paper is

Is there a unifying framework that accommodates the many difficult isomorphism testing problems arising in practice?

Such a framework would help to explain the difficulties from various areas when dealing with these isomorphism problems, and facilitate dialogue among researchers from different fields.

At first sight, this seems quite difficult: these problems concern very different mathematical objects, ranging from sets of quadratic equations, to algebras, to finite groups, to tensors, and each of them has its own rich theory.

Despite these obstacles, our first main result shows that those problems in Sec. 2 arising in many fields – from computational group theory to cryptography to machine learning – are equivalent under polynomial-time reductions. In proving the first main result, the  $d$ -TENSOR ISOMORPHISM problem occupies a central position. This leads us to define the complexity class TI, consisting of problems reducible to TI, much in vein of the introduction of the GRAPH ISOMORPHISM complexity class GI [43].

---

<sup>1</sup> We thank James B. Wilson, who maintains a suite of algorithms for  $p$ -group isomorphism testing [16], for communicating this insight to us from his hands-on experience. We of course maintain responsibility for any possible misunderstanding, or lack of knowledge regarding the performance of other implemented algorithms.

► **Definition 1** (The  $d$ -TENSOR ISOMORPHISM problem).  $d$ -TENSOR ISOMORPHISM over a field  $\mathbb{F}$  is the problem: given two  $d$ -way arrays  $\mathbf{A} = (a_{i_1, \dots, i_d})$  and  $\mathbf{B} = (b_{i_1, \dots, i_d})$ , where  $i_k \in [n_k]$  for  $k \in [d]$ , and  $a_{i_1, \dots, i_d}, b_{i_1, \dots, i_d} \in \mathbb{F}$ , decide whether there are  $P_k \in \text{GL}(n_k, \mathbb{F})$  for  $k \in [d]$ , such that for all  $i_1, \dots, i_d$ ,

$$a_{i_1, \dots, i_d} = \sum_{j_1, \dots, j_d} b_{j_1, \dots, j_d} (P_1)_{i_1, j_1} (P_2)_{i_2, j_2} \cdots (P_d)_{i_d, j_d}. \quad (1)$$

Our first main result resolves an open question well-known to the experts:<sup>2</sup>

► **Theorem 2** (=Cor. A).  $d$ -TENSOR ISOMORPHISM reduces to 3-TENSOR ISOMORPHISM in time  $O(n^d)$ .

Thm. 2 is also key to the application to quantum information in Sec. 6.1.

Thus, while the 2TI problem is easy (it's just matrix rank), 3TI already captures the complexity of  $d$ TI for any  $d$ . This phenomenon is reminiscent of the transition in hardness from 2 to 3 in  $k$ -SAT,  $k$ -COLORING,  $k$ -MATCHING, and many other NP-complete problems. It is interesting that an analogous phenomenon – a transition to some sort of “universality” from 2 to 3 – occurs in the setting of isomorphism problems, which we believe are not NP-complete over finite fields (indeed, they cannot be unless PH collapses).

► **Definition 3** (TI). For any field  $\mathbb{F}$ ,  $\text{TI}_{\mathbb{F}}$  denotes the class of problems that are polynomial-time Turing (Cook) reducible to  $d$ -TENSOR ISOMORPHISM over  $\mathbb{F}$ , for some  $d$ . A problem is  $\text{TI}_{\mathbb{F}}$ -complete, if it is in  $\text{TI}_{\mathbb{F}}$ , and  $d$ -TENSOR ISOMORPHISM over  $\mathbb{F}$  for any  $d$  reduces to this problem.

By Thm. 2, we may take  $d = 3$  without loss of generality. When we write TI without mentioning the field, the result holds for any field.

## 3.2 TI-complete problems

Our second main result shows the wide applicability and robustness of the TI class.

► **Theorem 4** (Informal statement of part of Theorem B). All the problems mentioned in Sec. 2 are TI-hard: IP2S, TENSOR CONGRUENCE, CUBIC FORM EQUIVALENCE (over fields of characteristic not 2 or 3), ALGEBRA ISOMORPHISM for commutative, unital, associative algebras, and GROUP ISOMORPHISM for  $p$ -groups of class 2 and exponent  $p$  given by matrix generators over  $\mathbb{F}_{p^e}$ .

In combination with the results of [26], we conclude that they are in fact TI-complete.

► **Remark 5.** Our results allow us to mostly answer a question from Saxena's thesis [64, p. 86]. Namely, Agrawal & Saxena [1] gave a reduction from CUBIC FORM EQUIVALENCE to RING ISOMORPHISM for commutative, unital, associative algebras over  $\mathbb{F}$ , under the assumption that every element of  $\mathbb{F}$  has a cube root in  $\mathbb{F}$ . For finite fields  $\mathbb{F}_q$ , the only such fields are those for which  $q = p^{2e+1}$  and  $p \equiv 2 \pmod{3}$ , which is asymptotically half of all primes. As explained after the proof of [1, Thm. 5], the use of cube roots seems inherent in their reduction, and Saxena asked whether such a reduction could be done over arbitrary fields. Using our results in conjunction with [26], we get a new such reduction – very different from the previous one [1] – which works over any field of characteristic not 2 or 3.

<sup>2</sup> We asked several experts who knew of the question, but we were unable to find a written reference. Interestingly, Oldenburger [56] worked on what we would call  $d$ -TENSOR ISOMORPHISM as far back as the 1930s. We would be grateful for any prior written reference to the question of whether  $d$ TI reduces to 3TI.



Here, we would also like to point out that some of the polynomial-time equivalences in Thm. 4, though perhaps expected by some experts, were not *a priori* clear. To get a sense for the non-obviousness of the equivalences of problems in Theorem 4, let us postulate the following hypothetical question. Recall that two matrices  $A, B \in M(n, \mathbb{F})$  are called *equivalent* if there exist  $P, Q \in GL(n, \mathbb{F})$  such that  $P^{-1}AQ = B$ , and they are *conjugate* if there exists  $P \in GL(n, \mathbb{F})$  such that  $P^{-1}AP = B$ . Can we reduce testing MATRIX CONJUGACY to testing MATRIX EQUIVALENCE? Of course since they are both in P there is a trivial reduction; to avoid this, let us consider only reductions  $r$  which send a matrix  $A$  to a matrix  $r(A)$  such that  $A$  and  $B$  are conjugate iff  $r(A)$  and  $r(B)$  are equivalent. Nearly all reductions between isomorphism problems that we are aware of have this form (so-called “kernel reductions” [25]; cf. functorial reductions [4]). This turns out to be essentially impossible. The reason is that the equivalence class of a matrix is completely determined by its rank, while the conjugacy class of a matrix is determined by its rational canonical form. Among  $n \times n$  matrices there are only  $n + 1$  equivalence classes, but there are at least  $|\mathbb{F}|^n$  rational canonical forms, coming from the choice of minimal polynomial/companion matrix. Even when  $\mathbb{F}$  is a finite field, such a reduction would thus require an exponential increase in dimension, and when  $\mathbb{F}$  is infinite, such a reduction is impossible regardless of running time.

Nonetheless, one of our results is that for *linear spaces* of matrices (one form of 3-way arrays; see Sec. 5.1), conjugacy testing and equivalence testing are polynomial-time equivalent. We say two subspaces  $\mathcal{A}, \mathcal{B} \subseteq M(n, \mathbb{F})$  are *conjugate* if there exists  $P \in GL(n, \mathbb{F})$  such that  $P\mathcal{A}P^{-1} = \{PAP^{-1} : A \in \mathcal{A}\} = \mathcal{B}$ , and analogously for equivalence. This is in sharp contrast to the case of single matrices. In the above setting, it means that there exists a polynomial-time computable map  $\phi$  from  $M(n, \mathbb{F})$  to *subspaces of*  $M(s, \mathbb{F})$ , such that  $A, B$  are conjugate up to a scalar if and only if  $\phi(A), \phi(B) \leq M(s, \mathbb{F})$  are equivalent as matrix spaces. Such a reduction may not be clear at first sight.

### 3.3 The relation between TENSOR ISOMORPHISM and GRAPH ISOMORPHISM

After introducing the TI class, it is natural to compare this class with the corresponding class for GRAPH ISOMORPHISM, GI.

Already by using known reductions [26, 30, 34, 48, 58], GRAPH ISOMORPHISM and PERMUTATIONAL CODE EQUIVALENCE reduce to 3-TENSOR ISOMORPHISM. For the inverse direction, we have the following connection.

► **Corollary 6.** *Let  $\mathbf{A}$  and  $\mathbf{B}$  be two 3-tensors over  $\mathbb{F}_q$ , and let  $n$  be the sum of the lengths of all three sides. To decide whether  $\mathbf{A}$  and  $\mathbf{B}$  are isomorphic reduces to solving GI for graphs of size  $q^{O(n)}$ .*

Therefore, if GI is in P, then  $3TI_{\mathbb{F}_q}$  can be solved in  $q^{O(n)}$  time, where  $n$  is the sum of the lengths of all three sides. More generally, if  $GI \in \text{TIME}(2^{O(\log n)^c})$  then  $3TI_{\mathbb{F}_q} \in \text{TIME}(q^{O(n^c)})$ . The current value of  $c$  for GI is 3 [5] (see [35] for the analysis of  $c$ ); improving  $c$  to be less than 2 would improve over the current state of the art for both GPI and 3TI.

In Fig. 1 we summarize the relationships between GI, TI, and many more isomorphism testing problems.

## 4 An overview of proof strategies and techniques

### 4.1 The main new technique

Our main new technique, used to show the reduction from  $d$ TI to 3TI (Thm. 2=Thm. A), is a simultaneous generalization of our reduction from 3TI to ALGEBRA ISOMORPHISM and the technique Grigoriev used [29] to show that isomorphism in a certain restricted class of algebras is equivalent to GI. In brief outline: a 3-way array  $\mathbf{A}$  specifies the structure constants of an algebra with basis  $x_1, \dots, x_n$  via  $x_i \cdot x_j := \sum_k \mathbf{A}(i, j, k)x_k$ , and this is essentially how we use it in the reduction from 3TI to ALGEBRA ISOMORPHISM. For arbitrary  $d \geq 3$ , we would like to similarly use a  $d$ -way array  $\mathbf{A}$  to specify how  $d$ -tuples of elements in some algebra  $\mathcal{A}$  multiply. The issue is that for  $\mathcal{A}$  to be an algebra, our construction must still specify how *pairs* of elements multiply. The basic idea is to let pairs (and triples, and so on, up to  $(d-2)$ -tuples) multiply “freely” (that is, without additional relations), and then to use  $\mathbf{A}$  to rewrite any product of  $d-1$  generators as a linear combination of the original generators. While this construction as described already gives one direction of the reduction (if  $\mathbf{A} \cong \mathbf{B}$ , then  $\mathcal{A} \cong \mathcal{B}$ ), the other direction is trickier. For that, we modify the construction to an algebra in which short products (less than  $d-2$  generators) do not quite multiply freely, but almost. After the fact, we found out that this construction generalizes the one used by Grigoriev [29] to show that GI was equivalent ALGEBRA ISOMORPHISM for a certain restricted class of algebras (see Sec. 6 for a comparison).

### 4.2 The proof strategy for Theorem 4=B

Let us now explain briefly on the proof of Thm. B=Thm. 4. The first step is to realize all of these problems in a single unifying viewpoint. That is, all these equivalence relations underlying these isomorphism testing problems can be realized as the orbits of certain natural group actions by direct products of general linear groups on 3-way arrays. We shall explain this in detail in Sec. 5. Here, we only demonstrate five group actions on 3-way arrays, and indicate how those practical problems correspond to some of these actions.

To introduce these five group actions, it is instructive to first examine the more familiar cases of matrices. There are three natural group actions on  $M(n, \mathbb{F})$ : for  $A \in M(n, \mathbb{F})$ , (1)  $(P, Q) \in GL(n, \mathbb{F}) \times GL(n, \mathbb{F})$  sends  $A$  to  $P^t A Q$ , (2)  $P \in GL(n, \mathbb{F})$  sends  $A$  to  $P^{-1} A P$ , and (3)  $P \in GL(n, \mathbb{F})$  sends  $A$  to  $P^t A P$ . These three actions endow  $A$  with different algebraic/geometric interpretations: (1) a linear map from a vector space  $V$  to another vector space  $W$ , (2) a linear map from  $V$  to itself, and (3) a bilinear map from  $V \times V$  to  $\mathbb{F}$ .

The five group actions on 3-way arrays referred to above are precisely analogous to the matrix setting. For a 3-way array  $\mathbf{A} = (a_{i,j,k})$ ,  $i, j, k \in [n]$ ,  $a_{i,j,k} \in \mathbb{F}$ , these actions are (1)  $(P_1, P_2, P_3) \in GL(n, \mathbb{F}) \times GL(n, \mathbb{F}) \times GL(n, \mathbb{F})$  acts on  $\mathbf{A}$  according to Equation 1 with  $d=3$ ; (2)  $(P_1, P_2) \in GL(n, \mathbb{F}) \times GL(n, \mathbb{F})$  acts on  $\mathbf{A}$  as  $(P_1^{-t}, P_1, P_2)$  in (1), where  $P^{-t}$  denotes the transpose of the inverse of  $P$ ; (3)  $(P_1, P_2) \in GL(n, \mathbb{F}) \times GL(n, \mathbb{F})$  acts on  $\mathbf{A}$  as  $(P_1, P_1, P_2)$  in (1); (4)  $P \in GL(n, \mathbb{F})$  acts on  $\mathbf{A}$  as  $(P, P, P)$  in (1); and (5)  $P \in GL(n, \mathbb{F})$  acts on  $\mathbf{A}$  as  $(P, P, P^{-t})$  in (1).

These five actions endow various families of 3-way arrays with different algebraic/geometric meanings, including 3-tensors, bilinear maps, matrix (associative or Lie) algebras, and trilinear forms, a.k.a. non-commutative cubic forms. It is then not difficult to cast each of the problems in Thm. 4 as (a special case of) the problem of deciding whether two 3-way arrays are in the same orbit under one of the five group actions; see Sec. 5.1 for detailed explanations.<sup>3</sup>

The first step only provides the context for proving Thm. 4. After the first step, we need to devise polynomial-time reductions among those isomorphism testing problems for 3-way arrays under these five group actions, often with certain restrictions on the 3-way array structures. The two basic ideas for these reductions are a gadget construction from [26], and the “embedding” technique from [27]. To implement these ideas, however, usually involves detailed and complicated computations.

For example, in the proof of Theorem 4, we use a gadget construction from [26] for the reduction from TENSOR ISOMORPHISM to IP2S. To show that this gadget works in our setting, we need a proof strategy that is different from that in [26]. Furthermore, the gadget from [26] introduces a quadratic blow-up in the input parameters. We then devise a new gadget, which achieves the same function with only linear blow-up, and enables Corollary 6. Having only linear blow-up is important in applications, e.g., to GROUP ISOMORPHISM in the Cayley table model (see [33]).

## 5 More details and more results on TI-completeness

### 5.1 Five group actions on 3-way arrays and the corresponding mathematical objects

In Section 3, we briefly defined five group actions on 3-way arrays with the help of Equation 1. However, the formulas for these group actions on 3-way arrays are somewhat unwieldy; our experience suggests that they are more easily digested when presented in the context of some of the natural interpretations of 3-way arrays as mathematical objects, which will also allow us to connect them back to the problems of Section 2. To connect the interpretations with the formulas themselves, one technical tool is very useful, namely, given a 3-way array  $\mathbf{A}(i, j, k)$ , we define its *frontal slices* to be the matrices  $A_k$  defined by  $A_k(i, j) := \mathbf{A}(i, j, k)$ ; that is, we think of the box of  $\mathbf{A}$  as arranged so that the  $i$  and  $j$  axes lie in the page, while the  $k$ -axis is perpendicular to the page. Similarly, its *lateral slices* (viewing the 3D box of  $\mathbf{A}$  “from the side”) are defined by  $L_j(i, k) := \mathbf{A}(i, j, k)$ . An  $\ell \times n \times m$  3-way array thus has  $m$  frontal slices and  $n$  lateral slices.

A natural action on arrays of size  $\ell \times n \times m$  is that of  $\text{GL}(\ell, \mathbb{F}) \times \text{GL}(n, \mathbb{F}) \times \text{GL}(m, \mathbb{F})$  by change of basis in each of the 3 directions, namely

$$((P, Q, R) \cdot \mathbf{A})(i', j', k') = \sum_{i, j, k} \mathbf{A}(i, j, k) P_{ii'} Q_{jj'} R_{kk'}.$$

We will see several interpretations of this action below.

**3-tensors.** A 3-way array  $\mathbf{A}(i, j, k)$ , where  $i \in [\ell]$ ,  $j \in [n]$ , and  $k \in [m]$ , is naturally identified as a vector in  $\mathbb{F}^\ell \otimes \mathbb{F}^n \otimes \mathbb{F}^m$ . Letting  $\vec{e}_i$  denote the  $i$ th standard basis vector of  $\mathbb{F}^n$ , a standard basis of  $\mathbb{F}^\ell \otimes \mathbb{F}^n \otimes \mathbb{F}^m$  is  $\{\vec{e}_i \otimes \vec{e}_j \otimes \vec{e}_k\}$ . Then  $\mathbf{A}$  represents the vector  $\sum_{i, j, k} \mathbf{A}(i, j, k) \vec{e}_i \otimes \vec{e}_j \otimes \vec{e}_k$ .

<sup>3</sup> While problems in Thm. 4 only use three out of those five actions, the other two actions also lead to problems that arise naturally, including MATRIX ALGEBRA CONJUGACY from [18], MATRIX LIE ALGEBRA CONJUGACY from [30], and BILINEAR MAP ISOTOPIISM from [13]; see Sec. 5.1 and Sec. 6.

in  $\mathbb{F}^\ell \otimes \mathbb{F}^n \otimes \mathbb{F}^m$ . The natural action by  $\text{GL}(\ell, \mathbb{F}) \times \text{GL}(n, \mathbb{F}) \times \text{GL}(m, \mathbb{F})$  above corresponds to changes of basis of the three vector spaces in the tensor product. The problem of deciding whether two 3-way arrays are the same under this action is called 3-TENSOR ISOMORPHISM.<sup>4</sup> This problem has been studied as far back as the 1930s [56].

**Cubic forms, trilinear forms, and tensor congruence.** From a 3-way array  $\mathbf{A}$  we can also construct a cubic form (=homogeneous degree 3 polynomial)  $\sum_{i,j,k} \mathbf{A}(i,j,k)x_i x_j x_k$ , where  $x_i$  are formal variables. If we consider the variables as commuting – or, equivalently, if  $\mathbf{A}$  is symmetric, meaning it is unchanged by permuting its three indices – we get an ordinary cubic form; if we consider them as non-commuting, we get a trilinear form (or “non-commutative cubic form”). In either case, the natural notion of isomorphism here comes from the action of  $\text{GL}(n, \mathbb{F})$  on the  $n$  variables  $x_i$ , in which  $P \in \text{GL}(n, \mathbb{F})$  transforms the preceding form into  $\sum_{i,j,k} \mathbf{A}(i,j,k)(\sum_{i'} P_{ii'} x_{i'}) (\sum_{j'} P_{jj'} x_{j'}) (\sum_{k'} P_{kk'} x_{k'})$ . In terms of 3-way arrays, we get  $(P \cdot \mathbf{A})(i', j', k') = \sum_{i,j,k} \mathbf{A}(i,j,k) P_{ii'} P_{jj'} P_{kk'}$ . The corresponding isomorphism problems are called CUBIC FORM EQUIVALENCE (in the commutative case) and TRILINEAR FORM EQUIVALENCE. This is identical to the TENSOR CONGRUENCE problem from [59] (where they worked over  $\mathbb{R}$ ).

**Matrix spaces.** Given a 3-way array  $\mathbf{A}$ , it is natural to consider the linear span of its frontal slices,  $\mathcal{A} = \langle A_1, \dots, A_m \rangle$ , also called a *matrix space*. One convenience of this viewpoint is that the action of  $\text{GL}(m, \mathbb{F})$  becomes implicit: it corresponds to change of basis *within* the matrix space  $\mathcal{A}$ . This allows us to generalize the three natural equivalence relations on matrices to matrix spaces: (1) two  $\ell \times n$  matrix spaces  $\mathcal{A}$  and  $\mathcal{B}$  are *equivalent* if there exists  $(P, Q) \in \text{GL}(\ell, \mathbb{F}) \times \text{GL}(n, \mathbb{F})$  such that  $PAQ = \mathcal{B}$ , where  $PAQ := \{PAQ : A \in \mathcal{A}\}$ ; (2) two  $n \times n$  matrix spaces  $\mathcal{A}, \mathcal{B}$  are *conjugate* if there exists  $P \in \text{GL}(n, \mathbb{F})$  such that  $PAP^{-1} = \mathcal{B}$ ; and (3) they are *isometric* if  $PAP^t = \mathcal{B}$ . The corresponding decision problems, when  $\mathcal{A}$  is given by a basis  $A_1, \dots, A_d$ , are MATRIX SPACE EQUIVALENCE, MATRIX SPACE CONJUGACY, and MATRIX SPACE ISOMETRY, respectively.

**Isomorphism of quadratic polynomials with 2 secrets.** For a tuple of homogeneous quadratic polynomials (over a field of characteristic not 2)  $\vec{f} = (f_1, \dots, f_m)$ , we may encode  $f_i$  by a symmetric matrix  $F_i$  in the usual way – where  $f_i(x) = x^t F_i x$  – and thus obtain a tuple of (symmetric) matrices  $(F_1, \dots, F_m)$ . Since, in the IP2S problem, we are also allowed to take linear combinations of the  $f_i$  themselves, we see that the IP2S problem is equivalent to the MATRIX SPACE ISOMETRY problem for  $\langle F_1, \dots, F_m \rangle$ . Equivalently, the action of  $(P, Q) \in \text{GL}(m, \mathbb{F}) \times \text{GL}(n, \mathbb{F})$  is by  $F_i \mapsto \sum_j P_{ij} Q^t F_j Q$ .

**Finite  $p$ -groups.** If we consider the quadratic polynomials  $f_i$  as defining a (symmetric) bilinear map  $\mathbb{F}^n \times \mathbb{F}^n \rightarrow \mathbb{F}^m$ , we may generalize to see that (not necessarily symmetric) bilinear maps arise naturally in other areas, notably in group theory. For matrices  $A_k$  over  $\mathbb{F}_p$ ,  $p$  an odd prime, we may consider MATRIX SPACE ISOMETRY for the matrix space  $\langle A_1, \dots, A_m \rangle$ . Two bilinear maps that are essentially the same up to such basis changes are sometimes called pseudo-isometric [17].

<sup>4</sup> Some authors call this TENSOR EQUIVALENCE; we use “ISOMORPHISM” both because this is the natural notion of isomorphism for such objects, and because we will be considering many different equivalence relations on essentially the same underlying objects.

When the  $A_k$  are skew-symmetric, Baer's correspondence [9] gives a bijection between finite  $p$ -groups of class 2 and exponent  $p$ , that is, in which  $g^p = 1$  for all  $g$  and in which  $[G, G] \leq Z(G)$ , and their corresponding skew-symmetric bilinear maps  $G/Z(G) \times G/Z(G) \rightarrow [G, G]$ , given by  $(gZ(G), hZ(G)) \mapsto [g, h] = ghg^{-1}h^{-1}$ . Two such groups are isomorphic if and only if their corresponding bilinear maps are pseudo-isometric, if and only if, using the matrix space terminology, the matrix spaces they span are isometric.

**Bilinear maps.** If we generalize even further to bilinear maps  $U \times V \rightarrow W$ , we find that from an  $\ell \times n \times m$  3-way array  $\mathbf{A}$ , we can construct such a bilinear map (=system of  $m$  bilinear forms)  $f_{\mathbf{A}} : \mathbb{F}^{\ell} \times \mathbb{F}^n \rightarrow \mathbb{F}^m$ , sending  $(u, v) \in \mathbb{F}^{\ell} \times \mathbb{F}^n$  to  $(u^t A_1 v, \dots, u^t A_m v)^t$ , where the  $A_k$  are the frontal slices of  $\mathbf{A}$ . The group action defining MATRIX SPACE EQUIVALENCE is equivalent to the action of  $\text{GL}(\ell, \mathbb{F}) \times \text{GL}(n, \mathbb{F}) \times \text{GL}(m, \mathbb{F})$  on such bilinear maps. This problem was recently studied under the name “testing isotopism of bilinear maps” in [13], in the context of testing isomorphism of graded algebras.

**Algebras.** We may also consider a 3-way array  $\mathbf{A}(i, j, k)$ ,  $i, j, k \in [n]$ , as the structure constants of an algebra (which need not be associative, commutative, nor unital), say with basis  $x_1, \dots, x_n$ , and with multiplication given by  $x_i \cdot x_j = \sum_k \mathbf{A}(i, j, k)x_k$ , and then extended (bi)linearly. Here the natural notion equivalence comes from the action of  $\text{GL}(n, \mathbb{F})$  by change of basis on the  $x_i$ . Despite the seeming similarity of this action to that on cubic forms, it turns out to be quite different: given  $P \in \text{GL}(n, \mathbb{F})$ , let  $\tilde{x}' = P\tilde{x}$ ; then we have  $x'_i \cdot x'_j = (\sum_i P_{i'i}x_i) \cdot (\sum_j P_{j'j}x_j) = \sum_{i,j} P_{i'i}P_{j'j}x_i \cdot x_j = \sum_{i,j,k} P_{i'i}P_{j'j}\mathbf{A}(i, j, k)x_k = \sum_{i,j,k} P_{i'i}P_{j'j}\mathbf{A}(i, j, k) \sum_{k'} (P^{-1})_{kk'}x_{k'}$ . Thus  $\mathbf{A}$  becomes  $(P \cdot \mathbf{A})(i', j', k') = \sum_{i,j,k} \mathbf{A}(i, j, k) P_{i'i}P_{j'j}(P^{-1})_{kk'}$ . The inverse in the third factor here is the crucial difference between this case and that of cubic or trilinear forms above, similar to the difference between matrix conjugacy and matrix isometry. The corresponding isomorphism problem is called ALGEBRA ISOMORPHISM.

**Summary.** The isomorphism problems of the above structures all have 3-way arrays as the underlying object, but are determined by different group actions. It is not hard to see that there are essentially five group actions in total: 3-TENSOR ISOMORPHISM, MATRIX SPACE CONJUGACY, MATRIX SPACE ISOMETRY, TRILINEAR FORM EQUIVALENCE, and ALGEBRA ISOMORPHISM. It turns out that these cover all the natural isomorphism problems on 3-way arrays in which the group acting is a product of  $\text{GL}(n, \mathbb{F})$  (where  $n$  is the side length of the arrays); see the full version [32] for a detailed discussion.

## 5.2 Full statement of main results

► **Theorem A.** *For any fixed  $d \geq 1$ ,  $d$ -TENSOR ISOMORPHISM reduces to ALGEBRA ISOMORPHISM.*

Combined with the results of [26], this immediately gives:

► **Corollary A.** *For any fixed  $d \geq 1$ ,  $d$ -TENSOR ISOMORPHISM reduces to 3-TENSOR ISOMORPHISM.*

Given the viewpoint of Section 5.1 on the problems from Section 2, to show that they are equivalent, it is enough to show that the isomorphism problems for 3-way arrays corresponding to the five group actions are equivalent, where 3-way arrays may also need to satisfy certain structural constraints (e.g., the frontal slices are symmetric or skew-symmetric). This is the content of our second main result.

► **Theorem B.** 3-TENSOR ISOMORPHISM reduces to each of the following problems in polynomial time.

1. GROUP ISOMORPHISM for  $p$ -groups exponent  $p$  ( $g^p = 1$  for all  $g$ ) and class 2 ( $G/Z(G)$  is abelian) given by generating matrices over  $\mathbb{F}_{p^e}$ . Here we consider only  $3\text{TI}_{\mathbb{F}_{p^e}}$  where  $p$  is an odd prime.
2. MATRIX SPACE ISOMETRY, even for alternating or symmetric matrix spaces.
3. MATRIX SPACE CONJUGACY, and even the special cases:
  - a. MATRIX LIE ALGEBRA CONJUGACY, for solvable Lie algebras  $L$  of derived length 2.<sup>5</sup>
  - b. ASSOCIATIVE MATRIX ALGEBRA CONJUGACY.<sup>6</sup>
4. ALGEBRA ISOMORPHISM, and even the special cases:
  - a. ASSOCIATIVE ALGEBRA ISOMORPHISM, for algebras that are commutative and unital, or for algebras that are commutative and 3-nilpotent ( $abc = 0$  for all  $a, b, c \in A$ )
  - b. LIE ALGEBRA ISOMORPHISM, for 2-step nilpotent Lie algebras ( $[u, [v, w]] = 0 \forall u, v, w$ )
5. CUBIC FORM EQUIVALENCE and TRILINEAR FORM EQUIVALENCE.

The algebras in (3) are given by a set of matrices which linearly span the algebra, while in (4) they are given by structure constants (see “Algebras” in Sec. 5.1).

Since the main result of [26] reduces the problems in Theorem B to 3-TENSOR ISOMORPHISM (cf. [26, Rmk. 1.1]), we have:

► **Corollary B.** Each of the problems listed in Theorem B is TI-complete.<sup>7</sup>

► **Remark 7.** Here is a brief summary of what is known about the complexity of some of these problems. Over a finite field  $\mathbb{F}_q$ , these problems are in  $\text{NP} \cap \text{coAM}$ . For  $\ell \times n \times m$  3-way arrays, the brute-force algorithms run in time  $q^{O(\ell^2 + n^2 + m^2)}$ , as  $\text{GL}(n, \mathbb{F}_q)$  can be enumerated in time  $q^{\Theta(n^2)}$ . Note that polynomial-time in the input size here would be  $\text{poly}(\ell, n, m, \log q)$ . Over any field  $\mathbb{F}$ , these problems are in  $\text{NP}_{\mathbb{F}}$  in the Blum–Shub–Smale model. When the input arrays are over  $\mathbb{Q}$  and we ask for isomorphism over  $\mathbb{C}$  or  $\mathbb{R}$ , these problems are in PSPACE using quantifier elimination. By Koiran’s celebrated result on Hilbert’s Nullstellensatz, for equivalence over  $\mathbb{C}$  they are in AM assuming the Generalized Riemann Hypothesis [44]. When the input is over  $\mathbb{Q}$  and we ask for equivalence over  $\mathbb{Q}$ , it is unknown whether these problems are even decidable; classically this is studied under ALGEBRA ISOMORPHISM for associative, unital algebras over  $\mathbb{Q}$  (see, e. g., [2, 60]), but by Cor. B, the question of decidability is open for all of these problems.

Over finite fields, several of these problems can be solved efficiently when one of the side lengths of the array is small. For  $d$ -dimensional spaces of  $n \times n$  matrices, MATRIX SPACE CONJUGACY and ISOMETRY can be solved in  $q^{O(n^2)} \cdot \text{poly}(d, n, \log q)$  time: once we fix an element of  $\text{GL}(n, \mathbb{F}_q)$ , the isomorphism problem reduces to solving linear systems of equations. Less trivially, MATRIX SPACE CONJUGACY can be solved in time  $q^{O(d^2)} \cdot \text{poly}(d, n, \log q)$  and 3TI for  $n \times m \times d$  tensors in time  $q^{O(d^2)} \cdot \text{poly}(d, n, m, \log q)$ , since once we fix an element of  $\text{GL}(d, \mathbb{F}_q)$ , the isomorphism problem either becomes an instance of, or reduces to [38], MODULE ISOMORPHISM, which admits several polynomial-time algorithms [15, 22, 37, 67]. Finally, one can solve MATRIX SPACE ISOMETRY in time  $q^{O(d^2)} \cdot \text{poly}(d, n, \log q)$ : once one fixes an element of  $\text{GL}(d, \mathbb{F}_q)$ , there is a rather involved algorithm [38], which uses the  $*$ -algebra technique originated from the study of computing with  $p$ -groups [17, 69].

<sup>5</sup> And even further, where  $L/[L, L] \cong \mathbb{F}$ .

<sup>6</sup> Even for algebras  $A$  whose Jacobson radical  $J(A)$  squares to zero and  $A/J(A) \cong \mathbb{F}$ .

<sup>7</sup> For CUBIC FORM EQUIVALENCE, we only show that it is in  $\text{TI}_{\mathbb{F}}$  when  $\text{char } \mathbb{F} > 3$  or  $\text{char } \mathbb{F} = 0$ .



## 6 Implications, more related works, and further discussions

### 6.1 An implication to quantum information

Quantum information is the study of information-theoretic properties of quantum states and channels, such as entanglement, non-classical correlations, and the uses of quantum states and channels for various computational tasks. A pure quantum particle takes states in a Hilbert space (=complex vector space, along with an inner product)  $V$ ; a pure multi-particle system takes states in the tensor product of the corresponding Hilbert spaces  $V_1 \otimes V_2 \otimes \cdots \otimes V_k$ .

A fundamental relation between  $k$ -partite quantum states is that of equivalence under *stochastic local operations and classical communication* (SLOCC) [10, 23]. If we imagine each particle is held by a different party, a “local operation” is an operation that a single party  $i$  can perform on its state in  $V_i$ . Although the definition of SLOCC involves combining this with classical communication, an equivalent definition is that two  $k$ -particle states  $\psi, \phi \in V_1 \otimes \cdots \otimes V_k$  are SLOCC-equivalent if they are in the same orbit under the action of the product of general linear groups  $\text{GL}(V_1) \times \text{GL}(V_2) \times \cdots \times \text{GL}(V_k)$  [23].<sup>8</sup> Deciding SLOCC equivalence (of un-normalized quantum states) is thus precisely the same as TI.

In this light, we may interpret our Thm. A as saying that SLOCC equivalence classes for  $k$ -partite entanglement can be simulated by SLOCC equivalence classes of tripartite entanglement. This might at first seem surprising, since bipartite entanglement is much better understood than tripartite or higher entanglement, so one might naively expect that 4-partite entanglement should be more complicated than tripartite, and so on. Our results show that in fact the tripartite case is already universal. This may be compared with a recent result in [72], which gives a transformation of multipartite states to a *set* of tripartite or bipartite states, interrelated by a *tensor network*, whereas our reduction produces a single tripartite state.

### 6.2 Further related works

While most of the related works have already been introduced before, we collect some of the key ones here for further discussions and comparisons.

The most closely related work is that of Futorny, Grochow, and Sergeichuk [26]. They show that a large family of isomorphism problems on 3-way arrays – including those involving multiple 3-way arrays simultaneously, or 3-way arrays that are partitioned into blocks, or 3-way arrays where some of the blocks or sides are acted on by the same group (e. g., MATRIX SPACE ISOMETRY) – all reduce to 3TI. Our work complements theirs in that all our reductions for Thm. B go in the opposite direction, reducing 3TI to other problems. Furthermore, the resulting 3-way arrays from our reductions for Thm. B usually satisfy certain structural constraints, which allows for versatile mathematical interpretations. Some of our other results relate GI and CODE EQUIVALENCE to 3TI; the latter problems were not considered in [26]. Thm. A considers  $d$ -tensors for any  $d \geq 3$ , which were not considered in [26].

In [1, 2], Agrawal and Saxena considered CUBIC FORM EQUIVALENCE and testing isomorphism of commutative, associative, unital algebras. They showed that GI reduces to ALGEBRA ISOMORPHISM; COMMUTATIVE ALGEBRA ISOMORPHISM reduces to CUBIC FORM

---

<sup>8</sup> Some authors use the action by the product of *special* linear groups  $\text{SL}(V_i)$  instead, but the difference is actually that physicists typically consider *normalized* quantum states, which are elements in the corresponding projective space  $\mathbb{P}(V_1 \otimes \cdots \otimes V_k)$ . Because the difference between  $\text{SL}(V_i)$  and  $\text{GL}(V_i)$  is merely scalar matrices, and scalar matrices act trivially on projective space, the equivalence relation is the same.



EQUIVALENCE; and HOMOGENEOUS DEGREE- $d$  FORM EQUIVALENCE reduces to ALGEBRA ISOMORPHISM assuming that the underlying field has  $d$ th root for every field element. By combining a reduction from [26] and our main Theorem B, we get a new reduction from CUBIC FORM EQUIVALENCE to ALGEBRA ISOMORPHISM that works over any field in which  $3!$  is a unit, which is fields of characteristic 0 or  $p > 3$ .

There are several other works which consider related isomorphism problems. Grigoriev [29] showed that GI is equivalent to isomorphism of unital, associative algebras  $A$  such that the radical  $R(A)$  squares to zero and  $A/R(A)$  is abelian. Interestingly, we show TI-completeness for *conjugacy* of *matrix* algebras with the same abstract structure (even when  $A/R(A)$  is only 1-dimensional). Note the latter problem is equivalent to asking whether two representations of  $A$  are equivalent up to automorphisms of  $A$ . The proof of Thm. A uses algebras in which  $R(A)^d = 0$  when reducing from  $d$ TI; it also uses Grigoriev’s result in one step.

Brooksbank and Wilson [18] showed a reduction from ASSOCIATIVE ALGEBRA ISOMORPHISM (when given by structure constants) to MATRIX ALGEBRA CONJUGACY. Grochow [30], among other things, showed that GI and CODEEQ reduce to MATRIX LIE ALGEBRA CONJUGACY, which is a special case of MATRIX SPACE CONJUGACY.

In [42], Kayal and Saxena considered testing isomorphism of finite rings when the rings are given by structure constants. This problem generalizes testing isomorphism of algebras over finite fields. They put this problem in  $\text{NP} \cap \text{coAM}$  [42, Thm. 4.1], reduce GI to this problem [42, Thm. 4.4], and prove that counting the number of ring automorphism ( $\#RA$ ) is in  $\text{FP}^{\text{AM} \cap \text{coAM}}$  [42, Thm. 5.1]. They also present a ZPP reduction from GI to  $\#RA$ , and show that the decision version of the ring automorphism problem is in P.

### 6.3 Combinatorial and group-theoretic techniques for GI and TI

Comparing with GRAPH ISOMORPHISM also offers one way to see why isomorphism problems for 3-way arrays are difficult. Indeed, the techniques for GI face great difficulty when dealing with isomorphism problems for multi-way arrays. Recall that most algorithms for GI, including Babai’s [5], are built on two families of techniques: group-theoretic, and combinatorial. One of the main differences is that the underlying group action for GI is a permutation group acting on a combinatorial structure, whereas the underlying group actions for isomorphism problems for 3-way arrays are matrix groups acting on (multi)linear structures.

Already in moving from permutation groups to matrix groups, we find many new computational difficulties that arise naturally in basic subroutines used in isomorphism testing. For example, the membership problem for permutation groups is well-known to be efficiently solvable by Sims’s algorithm [68] (see, e.g., [65] for a textbook treatment), while for matrix groups this was only recently shown to be solvable with a number-theoretic oracle over finite fields of odd characteristic [6]. Correspondingly, when moving from combinatorial structures to (multi)linear algebraic structures, we also find severe limitation on the use of most combinatorial techniques, like individualizing a vertex. For example, it is quite expensive to enumerate all vectors in a vector space, while it is usually considered efficient to go through all elements in a set. Similarly, within a set, any subset has a unique complement, whereas within  $\mathbb{F}_q^n$ , a subspace can have up to  $q^{\Theta(n^2)}$  complements.

Given all the differences between the combinatorial and linear-algebraic worlds, it may be surprising that combinatorial techniques for GRAPH ISOMORPHISM can nonetheless be useful for GROUP ISOMORPHISM. Indeed, Li and Qiao [46] adapted the individualisation and refinement technique, as used by Babai, Erdős and Selkow [8], to tackle ALTERNATING

MATRIX SPACE ISOMETRY over  $\mathbb{F}_q$ . This algorithm was recently shown [14] to practically improve over the default algorithms in Magma [12]. However, this technique, though helpful to improve from the brute-force  $q^{n^2} \cdot \text{poly}(n, \log q)$  time, seems still limited to getting *average-case*  $q^{O(n)}$ -time algorithms.

## 6.4 Outlook

In light of Babai’s breakthrough on GI [5], it is natural to consider “what’s next?” for isomorphism problems. That is, what isomorphism problems stand as crucial bottlenecks to further improvements on GI, and what isomorphism problems should naturally draw our attention for further exploration? Of course, one of the main open questions in the area remains whether or not GI is in P. Babai [5, arXiv ver., Sec. 13.2 & 13.4] already lists several isomorphism problems for further study, including GROUP ISOMORPHISM, PERMUTATIONAL CODE EQUIVALENCE (of linear codes), and PERMUTATION GROUP CONJUGACY. The reader may see where these sit in Fig. 1.

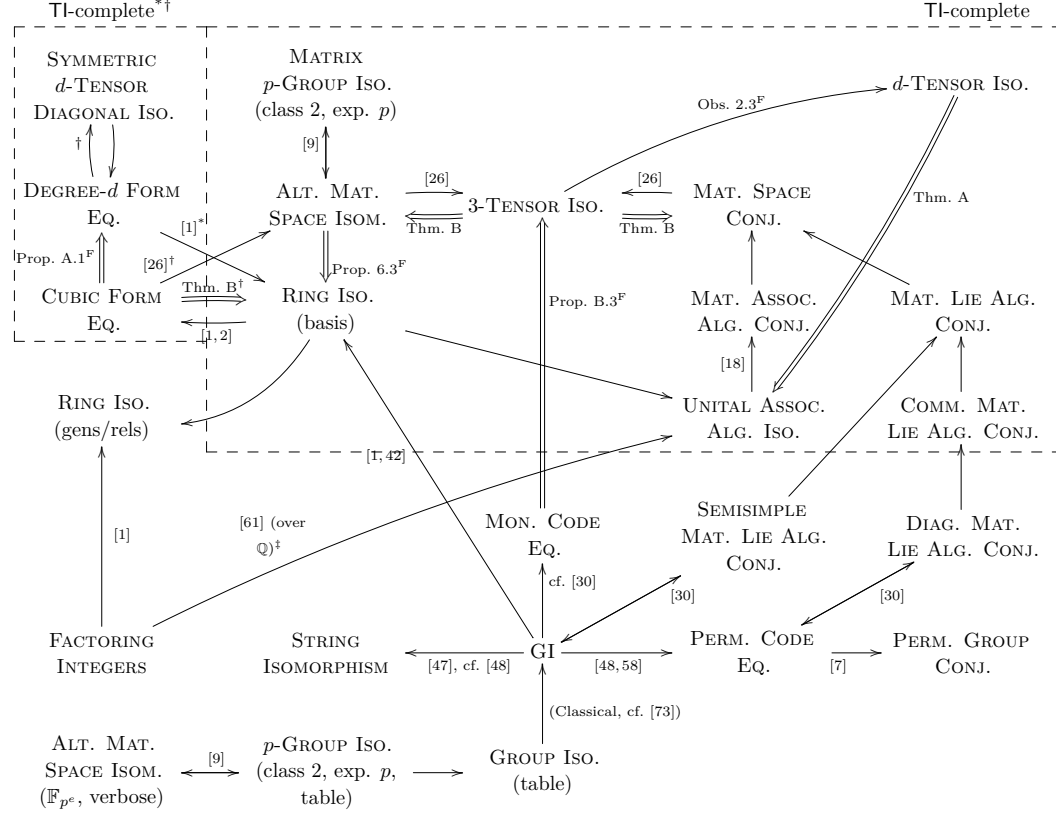
Based on the results above, we propose TI as a natural problem to study, both “after” GI, and to make further progress on GI itself. In particular, TI stands as a key bottleneck to put GI in P, because of the following. First, Babai suggested [5] that GROUP ISOMORPHISM (GPI) in the Cayley table model is a key bottleneck<sup>9</sup> to putting GI into P. Second, it has been long believed that  $p$ -groups of class 2 and exponent  $p$  are the hardest cases of GPI (for a number of reasons, see, e. g., [9, 36, 66, 71]). Third, by Baer’s correspondence [9], isomorphism for such groups is equivalent<sup>10</sup> to ALTERNATING MATRIX SPACE ISOMETRY (see Section 5.1). Finally, by our main Thm. B, ALTERNATING MATRIX SPACE ISOMETRY over  $\mathbb{F}_{p^e}$  is  $\text{TI}_{\mathbb{F}_{p^e}}$ -complete.

This then relates TI over finite fields to the believed-to-be-hardest instances of GPI, which in turn, as Babai suggested, is a key bottleneck for further progress on GI. We thus view the study of TI as a natural continuation of the study of GI. Furthermore, the main techniques for GI, namely the group-theoretic techniques and the combinatorial ones, also have corresponding techniques in the TI setting, although they are perhaps more complicated and less efficient than in the setting of GI. We explain this in detail in Sec. 6.

This theory for TI is far from complete, and many questions remain, largely inspired by the study of GI. In the full version [32, Section 10.1], we discuss a possible theory of universality for basis-explicit linear structures, in analogy with the universality of GI for explicit combinatorial structures [73, Section 15]. While not yet complete, this is another exciting reason to study TENSOR ISOMORPHISM and related problems, and it motivates some interesting open questions. Then we pose several natural open problems.

<sup>9</sup> Indeed, the current-best upper bounds on these two problems are now quite close:  $n^{O(\log n)}$  for GPI (originally due to [24, 53] – Miller attributes this to Tarjan – with improved constants [62, 63, 70]), and  $n^{O(\log^2 n)}$  for GI [5] (see [35] for calculation of the exponent).

<sup>10</sup> Specifically, solving ALTERNATING MATRIX SPACE ISOMETRY over  $\mathbb{F}_p$  in time  $p^{O(n+m)}$  is equivalent to testing isomorphism for  $p$ -groups of class 2 and exponent  $p$  in time polynomial in the group order, i.e. polynomial time in the Cayley table model.



■ **Figure 1** Summary of key isomorphism problems.  $A \rightarrow B$  indicates that  $A$  reduces to  $B$ , i. e.,  $A \leq_m^p B$ .  $A \Rightarrow B$  indicates a new result. Unattributed arrows indicate  $A$  is clearly a special case of  $B$ . Note that the definition of ring used in [1] is commutative, finite, and unital; by “algebra” we mean an algebra (not necessarily associative, let alone commutative nor unital) over a field. The reductions between RING ISO. (in the basis representation) and DEGREE- $d$  FORM EQ. and UNITAL ASSOCIATIVE ALGEBRA ISOMORPHISM are for rings over a field. The equivalences between ALTERNATING MATRIX SPACE ISOMETRY and  $p$ -GROUP ISOMORPHISM are for matrix spaces over  $\mathbb{F}_{p^e}$ . Some TI-complete problems from Thm. B are left out for clarity.

\* These results only hold over fields where every element has a  $d$ th root. In particular, DEGREE  $d$  FORM EQUIVALENCE and SYMMETRIC  $d$ -TENSOR ISOMORPHISM are TI-complete over fields with  $d$ -th roots. A finite field  $\mathbb{F}_q$  has this property if and only if  $d$  is coprime to  $q - 1$ .

† These results only hold over rings where  $d!$  is a unit.

‡ Assuming the Generalized Riemann Hypothesis, Rónyai [61] shows a Las Vegas randomized polynomial-time reduction from factoring square-free integers – probably not much easier than the general case – to isomorphism of 4-dimensional algebras over  $\mathbb{Q}$ . Despite the additional hypotheses, this is notable as the target of the reduction is algebras of *constant* dimension, in contrast to all other reductions in this figure.

<sup>F</sup> Refers to numbers in the full version [32].

## References

- 1 Manindra Agrawal and Nitin Saxena. Automorphisms of finite rings and applications to complexity of problems. In *STACS 2005, 22nd Annual Symposium on Theoretical Aspects of Computer Science, Proceedings*, pages 1–17, 2005. doi:10.1007/978-3-540-31856-9\_1.
- 2 Manindra Agrawal and Nitin Saxena. Equivalence of  $\mathbb{F}$ -algebras and cubic forms. In *STACS 2006, 23rd Annual Symposium on Theoretical Aspects of Computer Science, Proceedings*, pages 115–126, 2006. doi:10.1007/11672142\_8.
- 3 Eric Allender and Bireswar Das. Zero knowledge and circuit minimization. *Inf. Comput.*, 256:2–8, 2017. doi:10.1016/j.ic.2017.04.004.
- 4 László Babai. On the automorphism groups of strongly regular graphs I. In *Proceedings of the 5th Conference on Innovations in Theoretical Computer Science, ITCS '14*, pages 359–368, 2014. doi:10.1145/2554797.2554830.
- 5 László Babai. Graph isomorphism in quasipolynomial time [extended abstract]. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016*, pages 684–697, 2016. arXiv:1512.03547 [cs.DS] version 2. doi:10.1145/2897518.2897542.
- 6 László Babai, Robert Beals, and Ákos Seress. Polynomial-time theory of matrix groups. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009*, pages 55–64, 2009. doi:10.1145/1536414.1536425.
- 7 László Babai, Paolo Codenotti, Joshua A. Grochow, and Youming Qiao. Code equivalence and group isomorphism. In *Proceedings of the Twenty-Second Annual ACM–SIAM Symposium on Discrete Algorithms (SODA11)*, pages 1395–1408, Philadelphia, PA, 2011. SIAM. doi:10.1137/1.9781611973082.107.
- 8 László Babai, Paul Erdős, and Stanley M. Selkow. Random graph isomorphism. *SIAM J. Comput.*, 9(3):628–635, 1980. doi:10.1137/0209047.
- 9 Reinhold Baer. Groups with abelian central quotient group. *Trans. AMS*, 44(3):357–386, 1938. doi:10.1090/S0002-9947-1938-1501972-1.
- 10 Charles H. Bennett, Sandu Popescu, Daniel Rohrlich, John A. Smolin, and Ashish V. Thapliyal. Exact and asymptotic measures of multipartite pure-state entanglement. *Physical Review A*, 63(1):012307, 2000. doi:10.1103/PhysRevA.63.012307.
- 11 Jérémy Berthomieu, Jean-Charles Faugère, and Ludovic Perret. Polynomial-time algorithms for quadratic isomorphism of polynomials: The regular case. *J. Complexity*, 31(4):590–616, 2015. doi:10.1016/j.jco.2015.04.001.
- 12 W. Bosma, J. J. Cannon, and C. Playoust. The Magma algebra system I: the user language. *J. Symb. Comput.*, pages 235–265, 1997. doi:10.1006/jscs.1996.0125.
- 13 Peter Brooksbank, E O’Brien, and James Wilson. Testing isomorphism of graded algebras. *Trans. Amer. Math. Soc.*, 372:8067–8090, 2019. doi:10.1090/tran/7884.
- 14 Peter A. Brooksbank, Joshua A. Grochow, Yinan Li, Youming Qiao, and James B. Wilson. Incorporating Weisfeiler–Leman into algorithms for group isomorphism. arXiv:1905.02518 [cs.CC], 2019.
- 15 Peter A. Brooksbank and Eugene M. Luks. Testing isomorphism of modules. *J. Algebra*, 320(11):4020–4029, 2008. doi:10.1016/j.jalgebra.2008.07.014.
- 16 Peter A. Brooksbank, Joshua Maglione, and James B. Wilson. Thetensor.space. <https://github.com/thetensor-space/>, 2019.
- 17 Peter A. Brooksbank and James B. Wilson. Computing isometry groups of Hermitian maps. *Trans. Amer. Math. Soc.*, 364:1975–1996, 2012. doi:10.1090/S0002-9947-2011-05388-2.
- 18 Peter A Brooksbank and James B Wilson. The module isomorphism problem reconsidered. *Journal of Algebra*, 421:541–559, 2015. doi:10.1016/j.jalgebra.2014.09.004.
- 19 John Cannon and Derek F. Holt. Automorphism group computation and isomorphism testing in finite groups. *J. Symbolic Comput.*, 35(3):241–267, 2003. doi:10.1016/S0747-7171(02)00133-5.
- 20 Kuo-Tsai Chen. Integration of paths, geometric invariants and a generalized baker-hausdorff formula. *Annals of Mathematics*, pages 163–178, 1957. doi:10.2307/1969671.

- 21 Ilya Chevyrev and Andrey Kormilitzin. A primer on the signature method in machine learning, 2016. [arXiv:1603.03788](#).
- 22 Alexander Chistov, Gábor Ivanyos, and Marek Karpinski. Polynomial time algorithms for modules over finite dimensional algebras. In *Proceedings of the 1997 International Symposium on Symbolic and Algebraic Computation*, ISSAC '97, pages 68–74. ACM, 1997. doi:10.1145/258726.258751.
- 23 Wolfgang Dür, Guifre Vidal, and J. Ignacio Cirac. Three qubits can be entangled in two inequivalent ways. *Physical Review A*, 62(6):062314, 2000. doi:10.1103/PhysRevA.62.062314.
- 24 V. Felsch and J. Neubüser. On a programme for the determination of the automorphism group of a finite group. In Pergamon J. Leech, editor, *Computational Problems in Abstract Algebra (Proceedings of a Conference on Computational Problems in Algebra, Oxford, 1967)*, pages 59–60, Oxford, 1970.
- 25 Lance Fortnow and Joshua A. Grochow. Complexity classes of equivalence problems revisited. *Inform. and Comput.*, 209(4):748–763, 2011. Also available as [arXiv:0907.4775](#) [cs.CC]. doi:10.1016/j.ic.2011.01.006.
- 26 Vyacheslav Futorny, Joshua A. Grochow, and Vladimir V. Sergeichuk. Wildness for tensors. *Lin. Alg. Appl.*, 566:212–244, 2019. doi:10.1016/j.laa.2018.12.022.
- 27 I. M. Gelfand and V. A. Ponomarev. Remarks on the classification of a pair of commuting linear transformations in a finite-dimensional space. *Functional Anal. Appl.*, 3:325–326, 1969. doi:10.1007/BF01076321.
- 28 Daniel R. Grayson and Michael E. Stillman. Macaulay2, a software system for research in algebraic geometry. Available at <https://faculty.math.illinois.edu/Macaulay2/>.
- 29 D. Ju. Grigoriev. Complexity of “wild” matrix problems and of the isomorphism of algebras and graphs. *Zap. Nauchn. Sem. Leningrad. Otdel. Mat. Inst. Steklov. (LOMI)*, 105:10–17, 1981. Theoretical applications of the methods of mathematical logic, III. doi:10.1007/BF01084390.
- 30 Joshua A. Grochow. Matrix Lie algebra isomorphism. In *IEEE Conference on Computational Complexity (CCC12)*, pages 203–213, 2012. Also available as [arXiv:1112.2012](#) [cs.CC] and ECCC Technical Report TR11-168. doi:10.1109/CCC.2012.34.
- 31 Joshua A. Grochow and Youming Qiao. Algorithms for group isomorphism via group extensions and cohomology. *SIAM J. Comput.*, 46(4):1153–1216, 2017. Preliminary version in IEEE Conference on Computational Complexity (CCC) 2014 (DOI:10.1109/CCC.2014.19). Also available as [arXiv:1309.1776](#) [cs.DS] and ECCC Technical Report TR13-123. doi:10.1137/15M1009767.
- 32 Joshua A. Grochow and Youming Qiao. Isomorphism problems for tensors, groups, and cubic forms: completeness and reductions. *CoRR*, abs/1907.00309, 2019. [arXiv:1907.00309](#).
- 33 Joshua A. Grochow and Youming Qiao. On isomorphism problems for tensors, groups, and polynomials II: search and counting to decision reductions, and applications to group isomorphism, 2020. Under preparation.
- 34 Xiaoyu He and Youming Qiao. On the Baer–Lovász–Tutte construction of groups from graphs: isomorphism types and homomorphism notions, 2020. [arXiv:2003.07200](#) [math.CO].
- 35 Harald Andrés Helfgott, Jitendra Bajpai, and Daniele Dona. Graph isomorphisms in quasi-polynomial time. [arXiv:1710.04574](#) [math.GR], 2017.
- 36 Graham Higman. Enumerating  $p$ -groups. I. Inequalities. *Proc. London Math. Soc. (3)*, 10:24–30, 1960. doi:10.1112/plms/s3-10.1.24.
- 37 Gábor Ivanyos, Marek Karpinski, and Nitin Saxena. Deterministic polynomial time algorithms for matrix completion problems. *SIAM J. Comput.*, 39(8):3736–3751, 2010. doi:10.1137/090781231.
- 38 Gábor Ivanyos and Youming Qiao. Algorithms based on  $*$ -algebras, and their applications to isomorphism of polynomials with one secret, group isomorphism, and polynomial identity testing. *SIAM J. Comput.*, 48(3):926–963, 2019. doi:10.1137/18M1165682.

- 39 Zhengfeng Ji, Youming Qiao, Fang Song, and Aaram Yun. General linear group action on tensors: A candidate for post-quantum cryptography. In Dennis Hofheinz and Alon Rosen, editors, *Theory of Cryptography - 17th International Conference, TCC 2019, Nuremberg, Germany, December 1-5, 2019, Proceedings, Part I*, volume 11891 of *Lecture Notes in Computer Science*, pages 251–281. Springer, 2019. Preprint [arXiv:1906.04330](https://arxiv.org/abs/1906.04330) [cs.CR]. doi:10.1007/978-3-030-36030-6\_11.
- 40 Neeraj Kayal. Efficient algorithms for some special cases of the polynomial equivalence problem. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011, San Francisco, California, USA, January 23-25, 2011*, pages 1409–1421, 2011. doi:10.1137/1.9781611973082.108.
- 41 Neeraj Kayal. Affine projections of polynomials: extended abstract. In *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*, pages 643–662, 2012. doi:10.1145/2213977.2214036.
- 42 Neeraj Kayal and Nitin Saxena. Complexity of ring morphism problems. *Computational Complexity*, 15(4):342–390, 2006. doi:10.1007/s00037-007-0219-8.
- 43 Johannes Köbler, Uwe Schöning, and Jacobo Torán. *The graph isomorphism problem: its structural complexity*. Birkhauser Verlag, Basel, Switzerland, Switzerland, 1993. doi:10.1007/978-1-4612-0333-9.
- 44 Pascal Koiran. Hilbert’s Nullstellensatz is in the polynomial hierarchy. *J. Complexity*, 12(4):273–286, 1996. doi:10.1006/jcom.1996.0019.
- 45 J.M. Landsberg. *Tensors: Geometry and Applications*, volume 128 of *Graduate studies in mathematics*. American Mathematical Soc., 2012. doi:10.1090/gsm/128.
- 46 Yinan Li and Youming Qiao. Linear algebraic analogues of the graph isomorphism problem and the Erdős–Rényi model. In Chris Umans, editor, *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017*, pages 463–474. IEEE Computer Society, 2017. doi:10.1109/FOCS.2017.49.
- 47 Eugene M. Luks. Isomorphism of graphs of bounded valence can be tested in polynomial time. *J. Comput. Syst. Sci.*, 25(1):42–65, 1982. doi:10.1016/0022-0000(82)90009-5.
- 48 Eugene M. Luks. Permutation groups and polynomial-time computation. In *Groups and computation (New Brunswick, NJ, 1991)*, volume 11 of *DIMACS Ser. Discrete Math. Theoret. Comput. Sci.*, pages 139–175. Amer. Math. Soc., Providence, RI, 1993.
- 49 T. J. Lyons. Rough paths, signatures and the modelling of functions on streams. In *Proc. International Congress of Mathematicians*, pages 163–184. Kyung Moon Publishers, 2014.
- 50 Terry J. Lyons and Weijun Xu. Inverting the signature of a path. *J. Eur. Math. Soc.(JEMS)*, 20(7):1655–1687, 2018. doi:10.4171/JEMS/796.
- 51 Brendan D. McKay. Practical graph isomorphism. *Congr. Numer.*, pages 45–87, 1980.
- 52 Brendan D. McKay and Adolfo Piperno. Practical graph isomorphism, II. *Journal of Symbolic Computation*, 60(0):94–112, 2014. doi:10.1016/j.jsc.2013.09.003.
- 53 Gary L. Miller. On the  $n^{\log n}$  isomorphism technique (a preliminary report). In *STOC*, pages 51–58. ACM, 1978. doi:10.1145/800133.804331.
- 54 P. J. Moore, T. J. Lyons, and J. Gallacher. Using path signatures to predict a diagnosis of alzheimer’s disease. *PLoS ONE*, 14(9), 2019. doi:10.1371/journal.pone.0222212.
- 55 Eamonn A O’Brien. Isomorphism testing for  $p$ -groups. *Journal of Symbolic Computation*, 17(2):133–147, 1994. doi:10.1006/jsc.1994.1007.
- 56 Rufus Oldenburger. Non-singular multilinear forms and certain  $p$ -way matrix factorizations. *Trans. Amer. Math. Soc.*, 39(3):422–455, 1936. doi:10.2307/1989760.
- 57 Jacques Patarin. Hidden fields equations (HFE) and isomorphisms of polynomials (IP): two new families of asymmetric algorithms. In *Advances in Cryptology - EUROCRYPT ’96, International Conference on the Theory and Application of Cryptographic Techniques, Saragossa, Spain, May 12-16, 1996, Proceeding*, pages 33–48, 1996. doi:10.1007/3-540-68339-9\_4.
- 58 Erez Petrank and Ron M. Roth. Is code equivalence easy to decide? *IEEE Trans. Inf. Theory*, 43(5):1602–1604, 1997. doi:10.1109/18.623157.



- 59 Max Pfeffer, Anna Seigal, and Bernd Sturmfels. Learning paths from signature tensors. *SIAM Journal on Matrix Analysis and Applications*, 40(2):394–416, 2019. arXiv:1809.01588. doi:10.1137/18M1212331.
- 60 Bjorn Poonen. Undecidable problems: a sampler. In *Interpreting Gödel*, pages 211–241. Cambridge Univ. Press, Cambridge, 2014. arXiv:1204.0299 [math.LO].
- 61 Lajos Rónyai. Zero divisors in quaternion algebras. *J. Algorithms*, 9(4):494–506, 1988. doi:10.1016/0196-6774(88)90014-4.
- 62 David J. Rosenbaum. Bidirectional collision detection and faster deterministic isomorphism testing. arXiv preprint arXiv:1304.3935 [cs.DS], 2013.
- 63 David J. Rosenbaum. Breaking the  $n^{\log n}$  barrier for solvable-group isomorphism. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1054–1073. SIAM, 2013. Preprint arXiv:1205.0642 [cs.DS].
- 64 Nitin Saxena. *Morphisms of rings and applications to complexity*. PhD thesis, Indian Institute of Technology, Kanpur, May 2006. URL: <https://www.cse.iitk.ac.in/users/nitin/papers/thesis.pdf>.
- 65 Ákos Seress. *Permutation group algorithms*, volume 152. Cambridge University Press, 2003. doi:10.1017/CB09780511546549.
- 66 V. V. Sergeichuk. The classification of metabelian  $p$ -groups. In *Matrix problems (Russian)*, pages 150–161. Akad. Nauk Ukrain. SSR Inst. Mat., Kiev, 1977.
- 67 Vladimir V. Sergeichuk. Canonical matrices for linear matrix problems. *Linear Algebra Appl.*, 317(1-3):53–102, 2000. doi:10.1016/S0024-3795(00)00150-6.
- 68 Charles C Sims. Some group-theoretic algorithms. In *Topics in algebra*, pages 108–124. Springer, 1978. doi:10.1007/BFb0103126.
- 69 James B. Wilson. Decomposing  $p$ -groups via Jordan algebras. *J. Algebra*, 322:2642–2679, 2009. doi:10.1016/j.jalgebra.2009.07.029.
- 70 James B. Wilson. 2014 conference on *Groups, Computation, and Geometry* at Colorado State University, co-organized by P. Brooksbank, A. Hulpke, T. Penttila, J. Wilson, and W. Kantor. Personal communication, 2014.
- 71 James B. Wilson. Surviving in the wilderness. Talk presented at the Sante Fe Institute Workshop on Wildness in Computer Science, Physics, and Mathematics, 2015.
- 72 SM Zangi, Jun-Li Li, and Cong-Feng Qiao. Quantum state concentration and classification of multipartite entanglement. *Physical Review A*, 97(1):012301, 2018. doi:10.1103/PhysRevA.97.012301.
- 73 V. N. Zemlyachenko, N. M. Korneenko, and R. I. Tyshkevich. Graph isomorphism problem. *J. Soviet Math.*, 29(4):1426–1481, May 1985. doi:10.1007/BF02104746.





# Bounds on the QAC<sup>0</sup> Complexity of Approximating Parity

Gregory Rosenthal 

University of Toronto, Canada

<https://www.cs.toronto.edu/~rosenthal/>

rosenthal@cs.toronto.edu

---

## Abstract

QAC circuits are quantum circuits with one-qubit gates and Toffoli gates of arbitrary arity. QAC<sup>0</sup> circuits are QAC circuits of constant depth, and are quantum analogues of AC<sup>0</sup> circuits. We prove the following:

- For all  $d \geq 7$  and  $\varepsilon > 0$  there is a depth- $d$  QAC circuit of size  $\exp(\text{poly}(n^{1/d}) \log(n/\varepsilon))$  that approximates the  $n$ -qubit parity function to within error  $\varepsilon$  on worst-case quantum inputs. Previously it was unknown whether QAC circuits of sublogarithmic depth could approximate parity regardless of size.
- We introduce a class of “mostly classical” QAC circuits, including a major component of our circuit from the above upper bound, and prove a tight lower bound on the size of low-depth, mostly classical QAC circuits that approximate this component.
- Arbitrary depth- $d$  QAC circuits require at least  $\Omega(n/d)$  multi-qubit gates to achieve a  $1/2 + \exp(-o(n/d))$  approximation of parity. When  $d = \Theta(\log n)$  this nearly matches an easy  $O(n)$  size upper bound for computing parity exactly.
- QAC circuits with at most two layers of multi-qubit gates cannot achieve a  $1/2 + \exp(-o(n))$  approximation of parity, even non-cleanly. Previously it was known only that such circuits could not cleanly compute parity exactly for sufficiently large  $n$ .

The proofs use a new normal form for quantum circuits which may be of independent interest, and are based on reductions to the problem of constructing certain generalizations of the cat state which we name “nekomata” after an analogous cat yōkai.

**2012 ACM Subject Classification** Theory of computation → Circuit complexity; Theory of computation → Quantum complexity theory; Theory of computation → Quantum complexity theory

**Keywords and phrases** quantum circuit complexity, QAC<sup>0</sup>, fanout, parity, nekomata

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.32

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/2008.07470>. See “v3” for the full version that most closely corresponds to the ITCS proceedings version.

**Funding** Supported by NSERC (PGS D).

**Acknowledgements** Thanks to Benjamin Rossman and Henry Yuen for introducing me to this problem, and for having several helpful discussions throughout the research and writing processes. Thanks to Srinivasan Arunachalam, Daniel Grier, Ian Mertz, Eric Rosenthal, and Rahul Santhanam for helpful discussions as well. Part of this work was done while the author was visiting the Simons Institute for the Theory of Computing. Circuit diagrams were made using the Quantikz package [12].

## 1 Introduction

### 1.1 Background

A central problem in computational complexity theory is to prove lower bounds on the nonuniform circuit size required to compute explicit boolean functions. Since this appears to be out of reach given current techniques, research in circuit complexity has instead focused on



© Gregory Rosenthal;

licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 32; pp. 32:1–32:20

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

proving lower bounds in restricted circuit classes. There are now many known lower bounds in *classical* circuit complexity, as well as in quantum *query* complexity, but comparatively few lower bounds are known in quantum circuit complexity, which is the subject of the current paper.

The study of quantum circuit complexity was initiated in large part by Green, Homer, Moore and Pollett [8], who defined quantum analogues of a number of classical circuit classes. One of the seemingly most restrictive quantum circuit classes that they defined is the class of QAC<sup>0</sup> circuits, consisting of constant-depth QAC circuits, where QAC circuits are quantum circuits with arbitrary one-qubit gates and generalized Toffoli gates of arbitrary arity. (More precisely,  $(n + 1)$ -ary generalized Toffoli gates are defined by  $|x, b\rangle \mapsto |x, b \oplus \bigwedge_{j=1}^n x_j\rangle$  for  $x = (x_1, \dots, x_n) \in \{0, 1\}^n, b \in \{0, 1\}$ .) This is analogous to the classical circuit class of AC<sup>0</sup> circuits, consisting of constant-depth AC circuits, where AC circuits are boolean circuits with NOT gates and unbounded-fanin AND and OR gates. Low-depth circuits are a model of fast parallel computation, and this is especially important for quantum circuits, because quantum computations need to be fast relative to the decoherence time of the qubits in order to avoid error.

One difference between AC and QAC circuits is that AC circuits are allowed fanout “for free”, i.e. the input bits to the circuit and the outputs of gates may all be used as inputs to arbitrarily many gates. The quantum analogue of this would be to compute the unitary “fanout” transformation  $U_F$ , defined by  $U_F|b, x_1, \dots, x_{n-1}\rangle = |b, x_1 \oplus b, \dots, x_{n-1} \oplus b\rangle$  for  $b, x_1, \dots, x_{n-1} \in \{0, 1\}$ , or at least to compute this in the case that we call “restricted fanout” in which  $x_1 = \dots = x_{n-1} = 0$ . QAC<sup>0</sup> circuits with fanout gates are called QAC<sub>f</sub><sup>0</sup> circuits, and can simulate arbitrary AC<sup>0</sup> circuits by using ancillae and restricted fanout to make as many copies as needed of the input bits and of the outputs of gates. In fact, QAC<sub>f</sub><sup>0</sup> circuits are *strictly* more powerful than AC<sup>0</sup> circuits, because QAC<sub>f</sub><sup>0</sup> circuits (even without generalized Toffoli gates) of polynomial size can also compute threshold functions [11, 14] whereas AC<sup>0</sup> circuits require exponential size to do so [10]. In contrast, little is known about the power of QAC<sup>0</sup> circuits and how it compares with that of AC<sup>0</sup> circuits.

Green et al. [8] observed that fanout can be computed by QAC circuits of logarithmic depth and linear size. This raises the question of whether QAC circuits of *sublogarithmic* depth can compute fanout, or at least restricted fanout, even if allowed *arbitrary* size. The same question can be asked about parity, which is a famous example of a function that requires exponential size to compute in AC<sup>0</sup> [10], and which is defined for quantum circuits as the unitary transformation  $U_{\oplus}$  such that  $U_{\oplus}|b, x\rangle = |b \oplus \bigoplus_{j=1}^{n-1} x_j, x\rangle$  for  $b \in \{0, 1\}, x = (x_1, \dots, x_{n-1}) \in \{0, 1\}^{n-1}$ . In fact, all of these questions are equivalent: Green et al. [8] proved that parity and fanout are equivalent up to conjugation by Hadamard gates, and that they reduce to restricted fanout with negligible blowups in size and depth (see the full paper for illustrations).

Recent work [7, 9] suggests that QAC<sub>f</sub><sup>0</sup> may be a physically realistic model of constant depth computation in certain quantum computing architectures (such as ion traps). As for QAC lower bounds, Fang, Fenner, Green, Homer and Zhang [5] proved that QAC circuits with  $a$  ancillae require depth at least  $\Omega(\log(n/(a + 1)))$  to compute the  $n$ -qubit parity and fanout functions, which is a nontrivial lower bound when  $a$  is  $o(n)$ . Bera [3] used a different approach to prove something slightly weaker than the  $a = 0$  case of this result. Finally, Padé, Fenner, Grier and Thierauf [13] proved that QAC circuits with two layers of generalized Toffoli gates cannot cleanly<sup>1</sup> compute 4-qubit parity or fanout, regardless of the number of ancillae. A survey of Bera, Green and Homer [4] discusses some of the aforementioned QAC lower bounds and QAC<sub>f</sub> upper bounds in greater detail.

<sup>1</sup> A clean computation is one in which the ancillae end in the all-zeros state.

## 1.2 Results and Selected Proof Overviews

### 1.2.1 Definitions of Complexity Measures

Call  $|\langle\psi|\varphi\rangle|^2$  the *fidelity* of states  $|\varphi\rangle$  and  $|\psi\rangle$ . We define the *size* of a QAC circuit to be the number of multi-qubit gates in it, and the *depth* of a QAC circuit to be the number of layers of multi-qubit gates in it. One motivation for not counting single-qubit gates, besides mathematical convenience, is that size and depth can be interpreted as measures of the reliability and computation time of a quantum circuit respectively, and in practice multi-qubit gates tend to be less reliable and take more time to apply as compared to single-qubit gates.

### 1.2.2 Reductions to and from Constructing Nekomata

Recall that Green et al. [8] proved that parity, fanout, and restricted fanout are all equivalent up to low-complexity QAC reductions. In Section 3 we make the more general observation that clean approximate and non-clean approximate versions of these problems are all equivalent in this sense. For brevity's sake, here in Section 1.2 we will only state immediate corollaries of these reductions insofar as they relate to our other results.

We also introduce another problem equivalent to parity, which all of our results about parity and fanout are proved via reductions to. The state  $\frac{1}{\sqrt{2}} \sum_{b=0}^1 |b^n\rangle$  is commonly called the *cat state* on  $n$  qubits, and we denote it by  $|\mathbb{K}_n\rangle$ . More generally, call a state  $|\nu\rangle$  an *n-nekomata* if  $|\nu\rangle = \frac{1}{\sqrt{2}} \sum_{b=0}^1 |b^n, \psi_b\rangle$  for some states  $|\psi_0\rangle, |\psi_1\rangle$  on any number of qubits (the word “nekomata” is also the name of two-tailed cats from Chinese and Japanese folklore), or equivalently if a standard-basis measurement of some  $n$  qubits of  $|\nu\rangle$  outputs all-zeros and all-ones each with probability  $1/2$ .

Call a QAC circuit  $C$  acting on any number of qubits a solution to the “ $p$ -approximate  $n$ -nekomata problem” if there exists an  $n$ -nekomata  $|\nu\rangle$  such that  $C|0\dots 0\rangle$  and  $|\nu\rangle$  have fidelity at least  $p$ . (There is no need to allow “ancillae” in this problem, because if  $|\nu\rangle$  is an  $n$ -nekomata then so is  $|\nu, \psi\rangle$  for any state  $|\psi\rangle$ .) Note that the identity circuit on  $n$  or more qubits trivially solves the  $1/2$ -approximate  $n$ -nekomata problem. In informal discussions we will often say that a circuit “constructs an approximate  $n$ -nekomata” if it solves the  $p$ -approximate  $n$ -nekomata problem for some fixed  $p \in (1/2, 1)$ , say  $p = 3/4$ .

Constructing nekomata reduces to computing restricted fanout because  $|\mathbb{K}_n\rangle = U_F(H \otimes I)|0^n\rangle$ . Our reduction from parity to constructing nekomata is a variant of Green et al.’s [8] reduction from parity to restricted fanout.

### 1.2.3 Upper Bounds

► **Theorem 1.1.** *For all  $\varepsilon > 0$  there exists a depth-2 QAC circuit  $C$  such that for some  $n$ -nekomata  $|\nu\rangle$ , the fidelity of  $C|0\dots 0\rangle$  and  $|\nu\rangle$  is at least  $1 - \varepsilon$ . Furthermore, the size of  $C$  and the number of qubits acted on by  $C$  are both  $\exp(O(n \log(n/\varepsilon)))$ .*

To state a stronger upper bound for approximating unitary transformations than can conveniently be done in terms of fidelity, call  $1 - \|\varphi - \psi\|_2^2$  the *phase-dependent fidelity* of states  $|\varphi\rangle$  and  $|\psi\rangle$ . This quantity is at most the fidelity of  $|\varphi\rangle$  and  $|\psi\rangle$  (Equation (2)).

► **Corollary 1.2.** *For all  $d \geq 7$  and  $\varepsilon > 0$  there exist depth- $d$  QAC circuits  $C_\oplus, C_F, C_{\mathbb{K}}$  of size and number of ancillae  $\exp(\text{poly}(n^{1/d}) \log(n/\varepsilon))$ , where the  $\text{poly}(n^{1/d})$  term is at most  $O(n)$ , such that for all  $n$ -qubit states  $|\phi\rangle$ ,*

- *the phase-dependent fidelity of  $C_\oplus|\phi, 0\dots 0\rangle$  and  $U_\oplus|\phi\rangle \otimes |0\dots 0\rangle$  is at least  $1 - \varepsilon$ ;*
- *the phase-dependent fidelity of  $C_F|\phi, 0\dots 0\rangle$  and  $U_F|\phi\rangle \otimes |0\dots 0\rangle$  is at least  $1 - \varepsilon$ ;*
- *the phase-dependent fidelity of  $C_{\mathbb{K}}|0\dots 0\rangle$  and  $|\mathbb{K}_n, 0\dots 0\rangle$  is at least  $1 - \varepsilon$ .*

The  $d = 11$  case of Corollary 1.2 follows immediately from Theorem 1.1 and our reduction from parity to constructing nekomata. We decrease the minimum depth from 11 to 7 using an optimization specific to the circuit from our proof of Theorem 1.1. We prove Corollary 1.2 for higher depths using the fact that  $n$ -qubit restricted fanout can be computed by a circuit consisting of  $d$  layers of  $n^{1/d}$ -qubit restricted fanout gates.

If we were to also count one-qubit gates toward size and depth, then statements similar to Theorem 1.1 and Corollary 1.2 would still hold, because without loss of generality a depth- $d$  QAC circuit acting on  $m$  qubits has at most  $d + 1$  layers of one-qubit gates and at most  $(d + 1)m$  one-qubit gates.

### 1.2.4 Tight Lower Bounds for Constructing Approximate Nekomata in “Mostly Classical” Circuits

Call a QAC circuit *mostly classical* if it can be written as  $CLML^\dagger$  (i.e.  $C$  is applied last) such that  $C$  consists only of generalized Toffoli gates,  $L$  is a layer of one-qubit gates, and  $M$  is a layer of generalized Toffoli gates. The circuit  $C$  here is a close analogue of (classical) AC circuits with bounded fanout, since generalized Toffoli gates can simulate classical AND and NOT gates. The following is apparent from our proof of Theorem 1.1:

► **Remark 1.3.** Theorem 1.1 remains true even if “QAC circuit” is replaced by “mostly classical QAC circuit”.

Motivated by Remark 1.3, we prove the following lower bound for constructing approximate nekomata in mostly classical circuits:

► **Theorem 1.4.** *Let  $C$  be a mostly classical circuit of size  $s$  and depth  $o(\log n)$ , acting on any number of qubits. Then for all  $n$ -nekomata  $|\nu\rangle$ , the fidelity of  $C|0 \dots 0\rangle$  and  $|\nu\rangle$  is at most  $1/2 + \exp(-n^{1-o(1)}/\max(\log s, \sqrt{n}))$ .*

(See Theorem 4.2 for a more precise tradeoff between depth and fidelity.) In particular, Theorem 1.4 implies that mostly classical circuits of depth  $o(\log n)$  require size at least  $\exp(n^{1-o(1)})$  to construct approximate  $n$ -nekomata, essentially matching the  $\exp(\tilde{O}(n))$  size upper bound from Theorem 1.1 and Remark 1.3. This lower bound does not contradict the  $\exp(n^{o(1)})$  size upper bounds of depth  $\omega(1)$  from Corollary 1.2, because our reductions between parity, fanout, and constructing nekomata do not in general map mostly classical circuits to mostly classical circuits. Since the identity circuit is mostly classical, the upper bound on the fidelity of  $C|0 \dots 0\rangle$  and  $|\nu\rangle$  in Theorem 1.4 is tight up to the value being exponentiated. Finally, if we also allow  $r$ -qubit parity and fanout gates in mostly classical circuits – a natural model for small values of  $r$ , in light of the upper bounds from Corollary 1.2 – then a trivial generalization of our proof of Theorem 1.4 implies that an identical statement holds for circuits of depth  $o(\log_{\max(r,2)} n)$ .

To prove Theorem 1.4, it suffices to prove that the Hamming weight of a standard-basis measurement of any  $n$  qubits of  $C|0 \dots 0\rangle$  is concentrated around some value. We use the fact that standard-basis measurements commute with generalized Toffoli gates, and, after some preparation, apply a concentration inequality of Gavinsky, Lovett, Saks and Srinivasan [6].

### 1.2.5 Lower Bounds for Arbitrary QAC Circuits of Low Size and Depth

Call the first  $n$  qubits of an  $n$ -nekomata  $\frac{1}{\sqrt{2}} \sum_{b=0}^1 |b^n, \psi_b\rangle$  the *targets* of that nekomata.

► **Theorem 1.5.** *There is a universal constant  $c > 0$  such that the following holds. Let  $C$  be a depth- $d$  QAC circuit acting on any number of qubits, and let  $|\nu\rangle$  be an  $n$ -nekomata such that at most  $cn/(d + 1)$  multi-qubit gates in  $C$  act on the targets of  $|\nu\rangle$ . Then the fidelity of  $C|0 \dots 0\rangle$  and  $|\nu\rangle$  is at most  $1/2 + \exp(-\Omega(n/(d + 1)))$ .*

► **Corollary 1.6.** *Let  $c$  be the constant from Theorem 1.5. Let  $C$  be a depth- $d$  QAC circuit acting on any number of qubits, and assume that, collectively, the first  $n$  of these qubits are acted on by at most  $cn/(d+1)$  multi-qubit gates in  $C$ . Then for all states  $|\psi\rangle$ ,*

- *for  $|\phi_\oplus\rangle = |0, +^{n-1}\rangle$ , the fidelity of  $C|\phi_\oplus, 0 \dots 0\rangle$  and  $U_\oplus|\phi_\oplus\rangle \otimes |\psi\rangle$  is at most  $1/2 + \exp(-\Omega(n/(d+1)))$ ;*
- *for  $|\phi_F\rangle = |+, 0^{n-1}\rangle$ , the fidelity of  $C|\phi_F, 0 \dots 0\rangle$  and  $U_F|\phi_F\rangle \otimes |\psi\rangle$  is at most  $1/2 + \exp(-\Omega(n/(d+1)))$ ;*
- *the fidelity of  $C|0 \dots 0\rangle$  and  $|\otimes_n, \psi\rangle$  is at most  $1/2 + \exp(-\Omega(n/(d+1)))$ .*

(Perhaps surprisingly, a sharp “phase change” near the  $cn/(d+1)$  threshold is in fact inherent to our proof. The  $+1$  in  $\exp(-\Omega(n/(d+1)))$  is necessary when  $C = H$  and  $|\nu\rangle = |+\rangle := \frac{|0\rangle+|1\rangle}{\sqrt{2}}$ .) For example, Theorem 1.5 implies that a depth-2 QAC circuit constructing an approximate  $n$ -nekomata must have at least  $\Omega(n)$  multi-qubit gates acting on the targets of that nekomata. This  $\Omega(n)$  lower bound is tight, because Theorem 1.1 says that depth-2 QAC circuits can construct approximate  $n$ -nekomata, and a depth- $d$  QAC circuit can have at most  $nd$  multi-qubit gates acting on any given set of  $n$  qubits. Similarly, Corollary 1.6 implies that depth-7 QAC circuits approximating  $n$ -qubit parity, fanout, or restricted fanout require at least  $\Omega(n)$  multi-qubit gates acting on the  $n$  “input” qubits, and this  $\Omega(n)$  lower bound is tight as well by Corollary 1.2.

Theorem 1.5 also implies that the *total* number of multi-qubit gates, a.k.a. the size, of a depth- $d$  QAC circuit constructing an approximate  $n$ -nekomata must be at least  $\Omega(n/(d+1))$ . When  $d$  is  $o(\log n)$ , this lower bound is disappointingly far from the upper bounds of Theorem 1.1 and Corollary 1.2. However, Green et al. [8] observed that for some  $d = \Theta(\log n)$ , a depth- $d$  QAC circuit of size  $O(n)$  can construct an  $n$ -nekomata (specifically, the  $n$ -qubit cat state), so for this value of  $d$  our  $\Omega(n/d)$  size lower bound is tight to within a logarithmic factor. Similarly, for some  $d = \Theta(\log n)$ , the minimum size of a depth- $d$  QAC circuit that approximates  $n$ -qubit parity, fanout, or restricted fanout is between  $\Omega(n/\log n)$  and  $O(n)$ , by Corollary 1.6 and upper bounds of Green et al.

If a QAC circuit has size  $s \leq o(\sqrt{n})$  then its depth  $d$  satisfies  $d \leq s \leq o(\sqrt{n})$ , so  $s \leq o(\sqrt{n}) \leq o(n/(d+1))$ . It follows from Theorem 1.5 and Corollary 1.6 that QAC circuits of *arbitrary* depth require size at least  $\Omega(\sqrt{n})$  to construct approximate  $n$ -nekomata, or to approximately compute  $n$ -qubit parity, fanout, or restricted fanout.<sup>2</sup>

Finally, we remark that Theorem 1.5 is actually a special case of a more general result, Theorem 5.2, about states  $|\psi\rangle$  such that for some orthogonal projections<sup>3</sup>  $Q_1, \dots, Q_n$  on arbitrary numbers of qubits,  $\langle\psi|(\bigotimes_{j=1}^n Q_j \otimes I)|\psi\rangle = \langle\psi|(\bigotimes_{j=1}^n (I - Q_j) \otimes I)|\psi\rangle = 1/2$ . (For example,  $n$ -nekomata satisfy this criterion with  $Q_j = |0\rangle\langle 0|$  for all  $j$ .) We will comment on this generalization of Theorem 1.5 again in Section 1.2.7.

## 1.2.6 A Normal Form for Quantum Circuits

Integral to our proof of Theorem 1.5 is a certain normal form for QAC circuits, which may be of independent interest since the standard quantum circuit model is that of QAC circuits whose gates have maximum arity 2. Here we give the underlying intuition, by way of analogy with well-known facts from classical circuit complexity. If we define AC circuits as consisting only of AND and NOT gates, then it cannot in general be assumed that the NOT gates are

<sup>2</sup> In the full paper we generalize this argument to hold for the number of multi-qubit gates acting on the target/input qubits.

<sup>3</sup> I.e.  $Q_j = Q_j^2 = Q_j^\dagger$  for all  $j$ .

all adjacent to the inputs. However, by DeMorgan's laws we may equivalently allow OR gates in AC circuits as well, and then it *can* be assumed that the NOT gates are all adjacent to the inputs.<sup>4</sup> Similarly, we introduce a certain further generalization of generalized Toffoli gates which allows us to assume that the one-qubit gates in a QAC circuit are all adjacent to the input.

### 1.2.7 Depth-2 Lower Bounds

► **Theorem 1.7.** *Let  $C$  be a depth-2 QAC circuit of arbitrary size, acting on any number of qubits. Then for all states  $|\psi\rangle$ ,*

- (i) *for  $|\phi_\oplus\rangle = |0, +^{n-1}\rangle$ , the fidelity of  $C|\phi_\oplus, 0 \dots 0\rangle$  and  $U_\oplus|\phi_\oplus\rangle \otimes |\psi\rangle$  is at most  $1/2 + \exp(-\Omega(n))$ ;*
- (ii) *for  $|\phi_F\rangle = |+, 0^{n-1}\rangle$ , the fidelity of  $C|\phi_F, 0 \dots 0\rangle$  and  $U_F|\phi_F\rangle \otimes |\psi\rangle$  is at most  $1/2 + \exp(-\Omega(n))$ ;*
- (iii) *the fidelity of  $C|0 \dots 0\rangle$  and  $|\boxtimes_n, \psi\rangle$  is at most  $1/2 + \exp(-\Omega(n))$ .*

Our proof of Theorem 1.7 gives a multiplicative constant of roughly  $1/10^{60000}$  implicit in the  $\Omega(\cdot)$  notation in the above inequalities, which makes them trivial for small values of  $n$ . If  $n$  is sufficiently large however, then Theorem 1.7 implies that depth-2 QAC circuits cannot approximate  $n$ -qubit parity, fanout, or restricted fanout, or approximately construct the  $n$ -qubit cat state, even if these approximations are not required to be clean. Still taking  $n$  to be sufficiently large, this improves on the previously mentioned result of Padé et al. [13] that depth-2 QAC circuits cannot cleanly compute parity exactly on four or more qubits.

Theorem 1.7 and Corollary 1.2 imply that for all sufficiently large  $n$ , the minimum depth of a QAC circuit approximating  $n$ -qubit parity is between 3 and 7 inclusive, and likewise for fanout, restricted fanout, and constructing the cat state. By Theorem 1.1 there *is* a depth-2 QAC circuit that constructs an approximate  $n$ -nekomata for all  $n$ , so any proof of Theorem 1.7 must use some property of  $|\boxtimes_n, \psi\rangle$  that does not hold for an arbitrary  $n$ -nekomata. Ours uses a property similar to the fact that if we measure some of the qubits in the “ $B$ ” register of  $|\boxtimes_n\rangle_A \otimes |\psi\rangle_B$  in an arbitrary basis, then the resulting state in registers  $A$  and  $B$  is still an  $n$ -nekomata.

Our proof of Theorem 1.7 mostly uses different techniques than those of Padé et al. An exception is the observation, of which we use a generalization, that if we define a “generalized  $Z$  gate” on any number of qubits by  $Z = I - 2|1 \dots 1\rangle\langle 1 \dots 1|$  then  $Z|0, \phi\rangle = |0, \phi\rangle$  and  $Z|1, \phi\rangle = |1\rangle \otimes Z|\phi\rangle$  for all states  $|\phi\rangle$ . We also incorporate a variant of the proof given by Bene Watts, Kothari, Schaeffer and Tal [2, Theorem 16] that there is no QNC circuit (QAC circuit whose gates have maximum arity 2) of depth  $o(\log n)$  that maps  $|0 \dots 0\rangle$  to  $|\boxtimes_n, 0 \dots 0\rangle$ : Using a “light cone” argument they prove that out of any  $n$  output qubits, there are at least two whose standard-basis measurements would be independent, but the standard-basis measurements of any two qubits in  $|\boxtimes_n\rangle$  are dependent.

Our proof of Theorem 1.7 goes roughly as follows. If there are only  $o(n)$  multi-qubit gates acting on the  $n$  targets of  $|\boxtimes_n, \psi\rangle$  then the result follows from Theorem 1.5. Otherwise, out of the multi-qubit gates acting on the targets, the average gate acts on  $O(1)$  targets, as would be the case in a QNC circuit. Using a variant of a light cone argument, we choose  $\Theta(n)$  pairwise disjoint sets of qubits on which to define orthogonal projections, and apply the generalization of Theorem 1.5 that was mentioned at the end of Section 1.2.5.

<sup>4</sup> Invoking this assumption results in a constant-factor blowup in size and no blowup in depth, where (as is customary) we do not count NOT gates toward the size or depth of AC circuits.



### 1.3 Organization

In Section 1.4 we introduce some miscellaneous notation and definitions. In Section 2 we give multiple equivalent characterizations of QAC circuits, including the previously mentioned normal form, and introduce some related definitions which we will use in more general contexts as well. In Section 3 we give reductions between parity, fanout, restricted fanout, and constructing nekomata; we also use these reductions to prove that the  $d \geq 11$  case of Corollary 1.2 follows from Theorem 1.1 (the  $d < 11$  case is proved in the full paper), that Corollary 1.6 follows from Theorem 1.5, and that Theorems 1.7(i) and 1.7(ii) follow from Theorem 1.7(iii). In Section 4 we prove our upper and lower bounds for constructing approximate nekomata in mostly classical circuits, Theorems 1.1 and 1.4. In Section 5 we prove our other main results, Theorem 1.5 and Theorem 1.7(iii). Sections 3 to 5 may be read in any order.

### 1.4 Preliminaries

We write  $\log$  and  $\ln$  to denote the logarithms base 2 and  $e$  respectively, and  $(x_j)_j$  to denote the tuple of all  $x_j$  for  $j$  in some implicit index set. Also let  $[n] = \{1, \dots, n\}$  and  $\|\psi\| = \sqrt{\psi^* \psi}$ , i.e.  $\|\cdot\|$  denotes the 2-norm. Anything written as  $\langle \cdot |$  or  $|\cdot\rangle$  is implicitly unit-length. “Proof sketch” environments are replaced by complete proofs in the full paper.

*Orthogonal projections* are linear transformations  $Q$  such that  $Q = Q^2 = Q^\dagger$ . For an orthogonal projection  $Q$  and a state  $|\varphi\rangle$ , we call  $\langle \varphi | Q | \varphi \rangle$  “the probability that  $|\varphi\rangle$  measures to  $Q$ ”. If  $Q = |\psi\rangle\langle\psi|$  then we also call this “the probability that  $|\varphi\rangle$  measures to  $|\psi\rangle$ ”, and it equals  $|\langle \psi | \varphi \rangle|^2$ , a.k.a. the *fidelity* of  $|\varphi\rangle$  and  $|\psi\rangle$ . More generally, if  $Q$  is an orthogonal projection on some Hilbert space  $\mathcal{H}$  then we call  $\langle \varphi | (Q \otimes I) | \varphi \rangle$  “the probability that the  $\mathcal{H}$  qubits of  $|\varphi\rangle$  measure to  $Q$ ”.

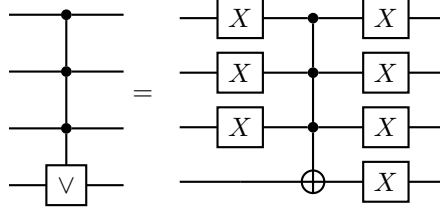
We use standard notation for the Hadamard basis states  $|+\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}$ ,  $|-\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}}$ , Hadamard gate  $H = |+\rangle\langle 0| + |-\rangle\langle 1|$ , and NOT gate  $X = |0\rangle\langle 1| + |1\rangle\langle 0|$ . We write  $I$  to denote the identity transformation,  $I_{\mathcal{H}}$  for the identity on the Hilbert space  $\mathcal{H}$ , and  $I_n$  for the identity on some  $n$ -qubit Hilbert space.

To be thorough, we remind the reader that an  $n$ -*nekomata* is a state with  $n$  qubits (called *targets*) that measure to  $0^n$  and to  $1^n$  each with probability  $1/2$ , or equivalently a state of the form  $\frac{1}{\sqrt{2}} \sum_{b=0}^1 |b^n, \psi_b\rangle$  for some states  $|\psi_0\rangle, |\psi_1\rangle$  on any number of qubits. For example, the  $n$ -qubit *cat state* is the state  $|\mathbb{E}_n\rangle = (|0^n\rangle + |1^n\rangle)/\sqrt{2}$ .

## 2 QAC Circuits

Consider a quantum circuit  $C$ , written as  $C = L_d M_d \cdots L_1 M_1 L_0$  such that each  $L_k$  consists only of one-qubit gates and each  $M_k$  is a layer (tensor product) of multi-qubit gates. We may assume that each  $L_k$  is a single layer as well, because the product of one-qubit gates is also a one-qubit gate. Define the *size* of  $C$  to be the number of multi-qubit gates in  $C$ , the *depth* of  $C$  to be the number of layers of multi-qubit gates in  $C$  (in this case,  $d$ ), and the *topology* of  $C$  to be the set of pairs  $(S, k)$  such that  $S$  equals the support of some gate in  $M_k$ , where the *support* of a gate is the set of qubits acted on by that gate. Note that the topology of  $C$  encodes its depth, size, and more generally the number of multi-qubit gates acting on any given set of qubits.

Recall that QAC circuits are quantum circuits with arbitrary one-qubit gates and generalized Toffoli gates of arbitrary arity, where  $(n+1)$ -ary generalized Toffoli gates are defined by  $|x, b\rangle \mapsto |x, b \oplus \bigwedge_{j=1}^n x_j\rangle$  for  $x = (x_1, \dots, x_n) \in \{0, 1\}^n$ ,  $b \in \{0, 1\}$ . Define an  $(n+1)$ -ary OR



■ **Figure 1** The multi-qubit gates on the left and right are OR and generalized Toffoli gates respectively, whose target qubits are on the bottom wire.

gate by  $|x, b\rangle \mapsto |x, b \oplus \bigvee_j x_j\rangle$  for  $x \in \{0, 1\}^n, b \in \{0, 1\}$ , and call the qubit corresponding to  $b$  in these definitions the *target* qubit of the gate. By the construction of an OR gate from a generalized Toffoli gate and NOT gates in Figure 1, we may add OR gates to the set of allowed gates when defining QAC circuits, without changing the set of topologies of QAC circuits computing any given unitary transformation.

For a state  $|\theta\rangle$  let  $R_{|\theta\rangle} = R_\theta = I - 2|\theta\rangle\langle\theta|$  (the R stands for “reflection”). Let a *mono-product state* be a tensor product of any number of one-qubit states. When  $|\theta\rangle$  is a mono-product state we call  $R_\theta$  an  $R_\otimes$  gate. For example, an  $(n+1)$ -qubit generalized Toffoli gate equals  $R_{|1^n, -\rangle}$ , because it acts on the basis  $\{|0\rangle, |1\rangle\}^{\otimes n} \otimes \{|+\rangle, |-\rangle\}$  by multiplying  $|1^n, -\rangle$  by -1 and leaving all other states in this basis unchanged.

Consider an  $(n+1)$ -qubit mono-product state  $|\theta\rangle$ , and let  $L$  be a layer of one-qubit gates such that  $|\theta\rangle = L|1^n, -\rangle$ . Then,

$$R_\theta = I - 2|\theta\rangle\langle\theta| = I - 2L|1^n, -\rangle\langle 1^n, -|L^\dagger = L(I - 2|1^n, -\rangle\langle 1^n, -|)L^\dagger = LR_{|1^n, -\rangle}L^\dagger, \quad (1)$$

i.e.  $R_\theta$  equals the conjugation of a generalized Toffoli gate by a layer of one-qubit gates. (Fang et al. [5] observed Equation (1) in the case where  $|\theta\rangle = |1^{n+1}\rangle$  and  $L = I_n \otimes H$ .) Therefore, similarly to the above, we may add arbitrary  $R_\otimes$  gates to the set of allowed gates when defining QAC circuits.

In fact, a stronger statement holds. Let a QAC circuit be in  $R_\otimes$  normal form if it can be written as  $CL$  such that  $C$  consists only of multi-qubit  $R_\otimes$  gates and  $L$  is a layer of single-qubit gates. We will use the following in Section 5:

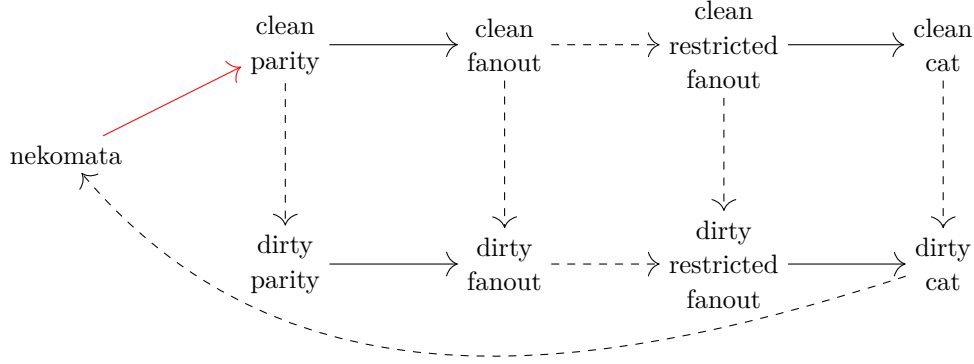
► **Proposition 2.1.** *Every QAC circuit computes the same unitary transformation as a circuit in  $R_\otimes$  normal form with the same topology.*

**Proof sketch.** The proof is by induction on the depth of the circuit. Let  $C = LMD$  be the circuit, where  $L$  (resp.  $M$ ) is the top layer of one-qubit (resp. multi-qubit) gates in  $C$ . Then write  $C = (LML^\dagger)(LD)$ , and apply Equation (1) and the inductive hypothesis. ◀

### 3 Reductions to and from Constructing Nekomata

In Section 3.1 we define the problems mentioned in the following theorem, and in Sections 3.1 and 3.2 we prove the second and first paragraphs of this theorem respectively:

► **Theorem 3.1.** *For all  $\varepsilon \geq 0$ , if there is a QAC circuit of size  $s$ , depth  $d$ , and number of qubits acted on  $a$  that solves the  $(1 - \varepsilon)$ -approximate  $n$ -nekomata problem, then there is a QAC circuit of size  $O(s + n)$ , depth  $4d + 3$ , and number of ancillae  $a$  that solves the  $(1 - O(\varepsilon))$ -approximate  $(n+1)$ -qubit clean parity problem.*



■ **Figure 2** A visualization of Theorem 3.1; see the theorem statement for the meaning of the arrows.

For all  $0 \leq p \leq 1$  and every non-red<sup>5</sup> arrow from a problem  $P$  to a problem  $Q$  in Figure 2, if a QAC circuit  $C$  solves  $p$ -approximate,  $n$ -qubit  $P$  then there is a QAC circuit with the same topology as  $C$  that solves  $p$ -approximate,  $n$ -qubit  $Q$ . (If  $Q$  is the nekomata problem then substitute “ $n$ -nekomata” for “ $n$ -qubit nekomata” here.) Furthermore, if this arrow is dashed then  $C$  itself solves  $p$ -approximate,  $n$ -qubit  $Q$ .

Then, using Theorem 3.1, in Section 3.3 we prove that the  $d \geq 11$  case of Corollary 1.2 follows from Theorem 1.1. It is easy to prove Corollary 1.6 assuming Theorem 1.5, and to prove Theorems 1.7(i) and 1.7(ii) assuming Theorem 1.7(iii), using reasoning similar to that in the proof of Theorem 3.1.

### 3.1 Problem Definitions and Most Reductions

We omit certain elementary parts of the proof, which may be found in the full paper. Recall that we define the *phase-dependent fidelity* of states  $|\varphi\rangle$  and  $|\psi\rangle$  to be  $1 - \|\varphi - \psi\|^2$ . This quantity is at most the fidelity of  $|\varphi\rangle$  and  $|\psi\rangle$ , because

$$|\langle\psi|\varphi\rangle|^2 \geq \left( \frac{\langle\psi|\varphi\rangle + \langle\varphi|\psi\rangle}{2} \right)^2 = \left( 1 - \frac{\|\varphi - \psi\|^2}{2} \right)^2 \geq 1 - \|\varphi - \psi\|^2. \quad (2)$$

► **Remark.** If  $\langle\varphi|\psi\rangle$  is a real number close to 1, say  $\langle\varphi|\psi\rangle = 1 - \varepsilon$ , then  $|\varphi\rangle$  and  $|\psi\rangle$  have fidelity  $1 - 2\varepsilon + \varepsilon^2$  and a nearly identical phase-dependent fidelity of  $1 - 2\varepsilon$ . On the other hand, if the phases of  $|\varphi\rangle$  and  $|\psi\rangle$  differ, then these states may have low phase-dependent fidelity even if their fidelity is close to 1.

The following two definitions are with respect to an arbitrary unitary transformation  $U$  on  $n$  qubits:

► **Problem 3.2** ( $p$ -approximate Clean  $U$ ). Construct a circuit  $C$  on at least  $n$  qubits such that for all  $n$ -qubit states  $|\phi\rangle$ , the phase-dependent fidelity of  $C|\phi, 0 \dots 0\rangle$  and  $U|\phi\rangle \otimes |0 \dots 0\rangle$  is at least  $p$ .

<sup>5</sup> Only the arrow from “nekomata” to “clean parity” is red.

► **Problem 3.3** (*p*-approximate Dirty  $U$ ). Construct a circuit  $C$  on at least  $n$  qubits such that for all  $n$ -qubit states  $|\phi\rangle$ , the first  $n$  qubits of  $C|\phi, 0 \dots 0\rangle$  measure to  $U|\phi\rangle$  with probability at least  $p$ .

Given  $n$ , recall that the unitary transformations for  $n$ -qubit parity and fanout are defined respectively by  $U_{\oplus}|b, x\rangle = |b \oplus \bigoplus_{j=1}^{n-1} x_j, x\rangle$  and  $U_F|b, x\rangle = |b, x_1 \oplus b, \dots, x_{n-1} \oplus b\rangle$  for  $b \in \{0, 1\}$ ,  $x = (x_1, \dots, x_{n-1}) \in \{0, 1\}^{n-1}$ . Define the clean and dirty versions of approximating  $n$ -qubit parity and fanout as instances of Problems 3.2 and 3.3 with respect to  $U_{\oplus}$  and  $U_F$ .

Green et al. [8] proved that  $H^{\otimes n} U_{\oplus} H^{\otimes n} = U_F$ . In the full paper we prove that if a circuit  $C$  computes  $p$ -approximate,  $n$ -qubit clean (resp. dirty) parity, then the circuit  $(H^{\otimes n} \otimes I)C(H^{\otimes n} \otimes I)$  computes  $p$ -approximate,  $n$ -qubit clean (resp. dirty) fanout.

► **Problem 3.4** (*p*-approximate Clean Restricted Fanout). Construct a circuit  $C$  on at least  $n$  qubits such that for all one-qubit states  $|\phi\rangle$ , the phase-dependent fidelity of  $C|\phi, 0^{n-1}, 0 \dots 0\rangle$  and  $U_F|\phi, 0^{n-1}\rangle \otimes |0 \dots 0\rangle$  is at least  $p$ .

► **Problem 3.5** (*p*-approximate Dirty Restricted Fanout). Construct a circuit  $C$  on at least  $n$  qubits such that for all one-qubit states  $|\phi\rangle$ , the first  $n$  qubits of  $C|\phi, 0^{n-1}, 0 \dots 0\rangle$  measure to  $U_F|\phi, 0^{n-1}\rangle$  with probability at least  $p$ .

► **Problem 3.6** (*p*-approximate Clean  $|\mathbb{B}_n\rangle$ ). Construct a circuit  $C$  on at least  $n$  qubits such that the phase-dependent fidelity of  $C|0 \dots 0\rangle$  and  $|\mathbb{B}_n, 0 \dots 0\rangle$  is at least  $p$ .

► **Problem 3.7** (*p*-approximate Dirty  $|\mathbb{B}_n\rangle$ ). Construct a circuit  $C$  on at least  $n$  qubits such that the first  $n$  qubits of  $C|0 \dots 0\rangle$  measure to  $|\mathbb{B}_n\rangle$  with probability at least  $p$ .

► **Problem 3.8** (*p*-approximate  $n$ -nekomata). Construct a circuit  $C$  such that for some  $n$ -nekomata  $|\nu\rangle$ , the fidelity of  $C|0 \dots 0\rangle$  and  $|\nu\rangle$  is at least  $p$ .

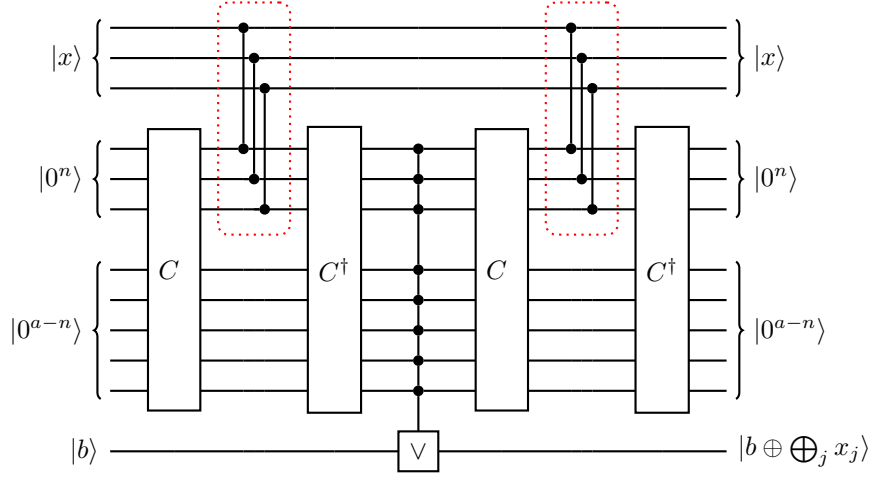
## 3.2 Reducing Clean Parity to Constructing Nekomata

Let  $C$  be a circuit on  $a$  qubits such that  $|\nu\rangle := C|0^a\rangle$  approximates an  $n$ -nekomata. A circuit for approximate  $(n+1)$ -qubit clean parity is shown in Figure 3, where the top  $n$  wires acted on by each of the  $C$  and  $C^\dagger$  subcircuits correspond to the targets of  $|\nu\rangle$ . Within each dotted rectangle is a layer of  $n$  copies of  $R_{|11\rangle}$ , the  $i$ 'th of which acts on the wires corresponding to the  $i$ 'th input qubit and the  $i$ 'th target of  $|\nu\rangle$  for  $i \in [n]$ . The gate  $R_{|11\rangle}$  is better known as a controlled  $Z$  gate, and acts as  $|xy\rangle \mapsto (-1)^{xy}|xy\rangle$  for  $x, y \in \{0, 1\}$ . In the middle of the circuit is an OR gate (recall Figure 1). The correctness of this circuit is proved in the full paper.

## 3.3 Proof of Corollary 1.2 ( $d \geq 11$ ) Assuming Theorem 1.1

► **Lemma 3.9** (essentially Green et al. [8]). *For all  $m \geq 2$  there is a quantum circuit of depth  $\lceil \log_m n \rceil$  and size at most  $n - 1$ , consisting only of restricted fanout gates of arity at most  $m$ , that computes  $n$ -qubit restricted fanout exactly using no ancillae.*

**Proof.** By linearity it suffices to consider input states of the form  $|b, 0^{n-1}\rangle$  for  $b \in \{0, 1\}$ . The proof is by induction on  $d = \lceil \log_m n \rceil$ , for a fixed value of  $m$ . Note that  $d - 1 < \log_m n \leq d$ , so  $m^{d-1} < n \leq m^d$ . If  $d = 0$  then  $n = 1$  and the identity circuit suffices. If  $d > 0$  then by induction we can map  $|b, 0^{n-1}\rangle$  to  $|b^{m^{d-1}}, 0^{n-m^{d-1}}\rangle$  in depth  $d - 1$  and size at most  $m^{d-1} - 1$ . Let  $n_1, \dots, n_{m^{d-1}} \in [m]$  be such that  $\sum_i n_i = n$ , and compute  $\bigotimes_i U_F|b, 0^{n_i-1}\rangle = |b^n\rangle$ . Since  $\bigotimes_i U_F$  has size at most  $n - m^{d-1}$  (omitting one-qubit gates), the total size of the circuit is at most  $(n - m^{d-1}) + (m^{d-1} - 1) = n - 1$ . ◀



■ **Figure 3** A circuit for parity, assuming  $C$  constructs an  $n$ -nekomata. See the surrounding text for further explanation.

Given Theorems 1.1 and 3.1 and Lemma 3.9, the proof of the  $d \geq 11$  case of Corollary 1.2 is elementary (if slightly tedious) and may be found in the full paper.

#### 4 Tight Bounds for Constructing Approximate Nekomata in “Mostly Classical” Circuits

Call a QAC circuit *purely classical* if it consists only of generalized Toffoli gates (including NOT gates, which are generalized Toffoli gates on one qubit). Call a QAC circuit *mostly classical* if it can be written as  $CL$  such that  $C$  is purely classical and  $L$  is a layer of  $R_\otimes$  gates; by Equation (1) this is equivalent to the definition from Section 1.2.4. Call a mostly classical QAC circuit *nice* if it can be written as  $CL$  in this way such that every multi-qubit gate  $R_\theta$  in  $L$  satisfies  $|\langle 0 \dots 0 | \theta \rangle|^2 \leq 1/4$ . (The niceness condition will allow us to express certain quantities as convex combinations in a convenient way, by ensuring that the coefficients in these convex combinations are between 0 and 1.) We prove the following generalizations of Theorems 1.1 and 1.4 respectively:

► **Theorem 4.1.** *For all  $2 \leq d \leq \log n$  and  $\varepsilon > 0$  there exists a nice, mostly classical, depth- $d$  QAC circuit  $C$  of size and number of qubits acted on  $\exp(O(n2^{-d} \log(n2^{-d}/\varepsilon))) + O(n)$  such that  $C|0 \dots 0\rangle$  has fidelity at least  $1 - \varepsilon$  with some  $n$ -nekomata.*

► **Theorem 4.2.** *Let  $C$  be a mostly classical circuit of size  $s$  and depth  $d$ .*

(i) *The fidelity of  $C|0 \dots 0\rangle$  and any  $n$ -nekomata is at most*

$$\frac{1}{2} + \exp\left(-\Omega\left(\frac{n/(4^d \log n)}{\max(\log s, \sqrt{n/(4^d \log n)})}\right)\right).$$

(ii) *If  $C$  is nice, then the fidelity of  $C|0 \dots 0\rangle$  and any  $n$ -nekomata is at most*

$$\frac{1}{2} + \exp\left(-\Omega\left(\frac{n/2^d}{\max(\log s, \sqrt{n/2^d})}\right)\right).$$

Theorems 4.1 and 4.2(ii) imply that for  $d \geq 2$ , the minimum size of a nice, mostly classical, depth- $d$  QAC circuit that “constructs an approximate  $n$ -nekomata” (i.e. maps  $|0 \dots 0\rangle$  to a state that has fidelity at least  $3/4$  with some  $n$ -nekomata) is between  $\exp(\Omega(n/2^d))$  and  $\exp(\tilde{O}(n/2^d)) + O(n)$ . We prove the  $d > 2$  case of Theorem 4.1 solely for the sake of comparison with Theorem 4.2(ii), as Theorem 4.1 gives a weaker upper bound than Corollary 1.2 when  $\omega(1) \leq d \leq o(\log n)$ . Theorem 4.2 makes a stronger statement about nice circuits than about non-nice circuits, since  $a/\max(\log s, \sqrt{a}) = \min(a/\log s, \sqrt{a})$  for all  $a > 0$ .

In Section 4.1 we make some general observations about mostly classical circuits and “approximate nekomata”, including observations common to the proofs of Theorems 4.1 and 4.2. In Section 4.2 we prove Theorem 4.1, and in Section 4.3 we prove Theorem 4.2(ii). We prove Theorem 4.2(i) in the full paper; its proof has a similar high-level idea to that of Theorem 4.2(ii), and is much more complicated.

## 4.1 Reduction to a Classical Sampling Problem

Collectively, the following observations reduce proving Theorems 4.1 and 4.2 to proving upper and lower bounds respectively for a certain type of sampling problem. This sampling problem can be succinctly characterized in purely classical and probabilistic terms, with only a transient reference to quantum circuits. Claims made in this subsection are proved in the full paper.

Recall that nekomata can be defined as states for which a standard-basis measurement of the targets is distributed in a certain way. The following two lemmas make similar statements about “approximate nekomata”, and are used to prove Theorems 4.1 and 4.2 respectively:

► **Lemma 4.3.** *Let  $|\varphi\rangle$  be a state with  $n$  “target” qubits that measure to all-zeros with probability exactly  $1/2$  and all-ones with probability at least  $1/2 - (2/3)\varepsilon$ . Then there exists an  $n$ -nekomata  $|\nu\rangle$  such that  $|\langle\nu|\varphi\rangle|^2 \geq 1 - \varepsilon$ .*

► **Lemma 4.4.** *Let  $|\varphi\rangle$  be a state with  $n$  “target” qubits that measure to all-zeros with probability  $p$  and all-ones with probability  $q$ . Then  $|\langle\nu|\varphi\rangle|^2 \leq 1/2 + \sqrt{\min(p, q)}$  for all  $n$ -nekomata  $|\nu\rangle$  with the same targets as  $|\varphi\rangle$ .*

Consider a mostly classical circuit, written as  $CL$  such that  $C$  is purely classical and  $L$  is a layer of  $R_{\otimes}$  gates. A standard-basis measurement of designated “target” qubits of  $CL|0 \dots 0\rangle$  is distributed identically to an appropriate marginal distribution of a standard-basis measurement of *all* qubits of  $CL|0 \dots 0\rangle$ . It is easy to see that standard-basis measurements commute with generalized Toffoli gates, so we may first measure  $L|0 \dots 0\rangle$  in the standard basis and then apply  $C$  to the result.

Finally, the following is straightforward to verify:

► **Lemma 4.5.** *Let  $(|\theta_j\rangle)_j$  be one-qubit states, and let  $p_j = |\langle 1|\theta_j\rangle|^2$  for all  $j$ . A standard-basis measurement of  $R_{\otimes_j |\theta_j\rangle}|0 \dots 0\rangle$  outputs all-zeros with probability  $\left(1 - 2\prod_j (1 - p_j)\right)^2$ , and any other boolean string  $(y_j)_j$  with probability  $4\prod_j (1 - p_j)P(\text{Bernoulli}(p_j) = y_j)$ .*

For mostly classical circuits that are nice, the following is a more convenient characterization of this distribution:

► **Corollary 4.6.** *If  $\prod_j (1 - p_j) \leq 1/4$  then the distribution from Lemma 4.5 is a convex combination of all-zeros with probability  $1 - 4\prod_j (1 - p_j)$  and  $(\text{Bernoulli}(p_j))_j$  with probability  $4\prod_j (1 - p_j)$ , where the  $\text{Bernoulli}(p_j)$  random variables are all independent.*

## 4.2 Proof of Theorem 4.1

Here we prove the depth-2 case of Theorem 4.1 (which is sufficient for all of our applications of Theorem 4.1); the generalization to depths greater than 2 is handled in the full paper.

► **Reminder** (depth-2 case of Theorem 4.1). *For all  $\varepsilon > 0$  there exists a nice, mostly classical, depth-2 QAC circuit  $C$  of size and number of qubits acted on  $\exp(O(n \log(n/\varepsilon)))$  such that  $C|0 \dots 0\rangle$  has fidelity at least  $1 - \varepsilon$  with some  $n$ -nekomata.*

**Proof.** Let  $M \in \mathbb{N}$  and  $\delta \in (0, 1)$  be parameters to be chosen later.<sup>6</sup> The circuit acts on  $n(M + 1)$  qubits, all initialized to  $|0\rangle$ , and arranged in a grid of dimensions  $n \times (M + 1)$  (Figure 4). Designate one column as the “target” column, and call the qubits in the  $M$  other columns “ancillae”. First, to each ancilla column, apply  $R_{(\sqrt{\delta}|0\rangle + \sqrt{1-\delta}|1\rangle)^{\otimes n}}$ . Second, to each row, apply an  $(M + 1)$ -qubit OR gate whose target qubit is in the target column. (A layer of OR gates is a depth-1 purely classical circuit, by the construction in Figure 1.)

All measurements described below are with respect to the state on the ancillae between the first and second layers of the above circuit. By Lemma 4.3 it suffices to choose  $M$  and  $\delta$  such that if we measure the ancillae in the standard basis, then with probability exactly  $1/2$  all of the ancillae measure to 0, and with probability at least  $1/2 - (2/3)\varepsilon$  at least one ancilla in each row measures to 1. We now choose  $\delta$  in terms of  $M$  such that the ancillae measure to all-zeros with probability  $1/2$ . By Lemma 4.5 and the independence of measurements of different columns, it suffices to ensure that  $(1 - 2\delta^n)^{2M} = 1/2$ . Choose  $\delta \in (0, (1/2)^{1/n})$  that satisfies this equation.

Let  $\varepsilon' = (2/3)\varepsilon$ . Below we will choose  $M$  such that the probability that there exists an ancilla column measuring to neither all-zeros nor all-ones is at most  $\varepsilon'$ . Equivalently, with probability at least  $1 - \varepsilon'$ , every ancilla column measures to either all-zeros or all-ones. Since the ancillae measure to all-zeros with probability  $1/2$ , it follows that with probability at least  $1/2 - \varepsilon'$ , every ancilla column measures to all-zeros or all-ones *and* at least one ancilla column measures to all-ones. Therefore the probability is at least  $1/2 - \varepsilon'$  that at least one ancilla in every row measures to 1, as desired.

By Lemma 4.5 and a union bound, the probability that there exists an ancilla column measuring to neither all-zeros nor all-ones is at most

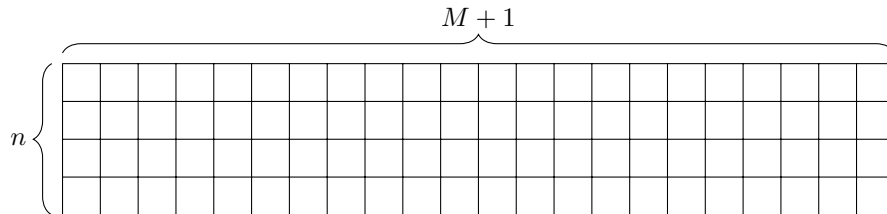
$$M(1 - (1 - 2\delta^n)^2 - 4\delta^n(1 - \delta)^n) = 4M\delta^n(1 - \delta^n - (1 - \delta)^n) \leq 4Mn\delta^{n+1}.$$

Since  $1/2 = (1 - 2\delta^n)^{2M} \leq \exp(-4\delta^n M)$ , it follows that  $\delta^n \leq \ln(2)/4M$ , so

$$4Mn\delta^{n+1} \leq 4Mn(\ln(2)/4M)^{1+1/n} = \ln(2)n(\ln(2)/4M)^{1/n}.$$

To make this bound at most  $\varepsilon'$ , let  $M = \lceil (\ln(2)/4) \cdot (\ln(2)n/\varepsilon')^n \rceil \leq \exp(O(n \log(n/\varepsilon)))$ . Finally, the circuit is nice because  $\delta^n \leq \ln(2)/4M \leq \ln(2)/4 < 1/4$ . ◀

<sup>6</sup> Ultimately we will let  $M = \exp(\Theta(n \log(n/\varepsilon)))$  and  $\delta = \Theta(\varepsilon/n)$ .



■ **Figure 4** The layout of the circuit.



### 4.3 Proof of Theorem 4.2(ii)

We use the following concentration inequality of Gavinsky, Lovett, Saks and Srinivasan [6]:

► **Definition 4.7** ([6]). *Call a random string  $(Y_1, \dots, Y_n) \in \{0, 1\}^n$  a read- $r$  family if there exist  $m \in \mathbb{N}$ , independent random variables  $X_1, \dots, X_m$ , sets  $S_1, \dots, S_n \subseteq [m]$  such that  $|\{j \mid i \in S_j\}| \leq r$  for all  $i \in [m]$ , and functions  $f_1, \dots, f_n$  such that  $Y_j = f_j((X_i)_{i \in S_j})$  for all  $j \in [n]$ .*

► **Theorem 4.8** ([6]). *Let  $(Y_1, \dots, Y_n)$  be a read- $r$  family, and let  $\mu = \mathbb{E}\left[\sum_{j=1}^n Y_j\right]$ . Then for all  $\varepsilon \geq 0$ ,*

$$\begin{aligned} P(Y_1 + \dots + Y_n \geq \mu + \varepsilon n) &\leq \exp(-2\varepsilon^2 n/r), \\ P(Y_1 + \dots + Y_n \leq \mu - \varepsilon n) &\leq \exp(-2\varepsilon^2 n/r). \end{aligned}$$

► **Remark.** For example, if  $r = 1$  then  $Y_1, \dots, Y_n$  are all independent and so Theorem 4.8 recovers a well-known Chernoff bound for sums of independent Bernoulli random variables. Theorem 4.8 also recovers this Chernoff bound when  $n = rm$  and  $Y_j = X_{\lceil j/r \rceil}$  for all  $j$  [6].

Consider a string  $x$  of independent Bernoulli random variables. If  $G$  is a generalized Toffoli gate then  $G|x\rangle$  is a read-2 family, because for all  $i$  the  $i$ 'th bit of  $x$  can only influence the  $i$ 'th and target bits of  $G|x\rangle$ . More generally, if  $G$  is a generalized Toffoli gate and  $L_1, L_2$  are layers of NOT gates acting on subsets of the support of  $G$ , then  $L_1 G L_2 |x\rangle$  is a read-2 family. Even more generally, it follows by induction that if  $C$  is a depth- $d$  purely classical circuit then  $C|x\rangle$  is a read- $2^d$  family.

Before proving Theorem 4.2(ii), as a warmup we briefly prove the following:

► **Proposition 4.9.** *If  $C$  is a depth- $d$  purely classical circuit and  $|\phi\rangle$  is a mono-product state, then  $|\langle \nu | C | \phi \rangle|^2 \leq 1/2 + \exp(-\Omega(n/2^d))$  for all  $n$ -nekomata  $|\nu\rangle$ .*

**Proof.** Since standard-basis measurements of qubits in a mono-product state are independent, it follows from the above discussion that a standard-basis measurement of any  $n$  designated target qubits of  $C|\phi\rangle$  is a read- $2^d$  family. If the expected Hamming weight of a standard-basis measurement of the targets of  $C|\phi\rangle$  is less (resp. greater) than or equal to  $n/2$ , then Theorem 4.8 implies that the targets of  $C|\phi\rangle$  measure to all-ones (resp. all-zeros) with probability at most  $\exp(-\Omega(n/2^d))$ , and the result follows from Lemma 4.4. ◀

► **Reminder** (Theorem 4.2(ii)). *If  $C$  is a nice, mostly classical circuit of size  $s$  and depth  $d$ , then the fidelity of  $C|0 \dots 0\rangle$  and any  $n$ -nekomata is at most*

$$\frac{1}{2} + \exp\left(-\Omega\left(\frac{n/2^d}{\max(\log s, \sqrt{n/2^d})}\right)\right).$$

**Abridged proof.** Designate  $n$  qubits of  $C|0 \dots 0\rangle$  as targets, and assume without loss of generality that  $s \geq \exp(\sqrt{n/2^d})$ . We will prove that for some  $a \in \{0, 1\}$ , the targets of  $C|0 \dots 0\rangle$  measure to  $a^n$  with probability at most  $\exp(-\Omega(n2^{-d}/\log s))$ . The result then follows from Lemma 4.4.

Write  $C = D(L \otimes_{G \in \mathcal{G}} G)$  such that  $D$  is purely classical,  $L$  is a layer of single-qubit gates, and  $\mathcal{G}$  is a set of multi-qubit  $R_\otimes$  gates that each satisfy the precondition of Corollary 4.6. For all  $G \in \mathcal{G}$ , a standard-basis measurement of  $G|0 \dots 0\rangle$  is distributed identically to  $(b_G \wedge x_{G,i})_i$  for some independent Bernoulli random variables  $b_G, (x_{G,i})_i$ , where  $\mathbb{E}[b_G] = 4 \prod_i (1 - \mathbb{E}[x_{G,i}])$ . Let  $\mu_G = \sum_i \mathbb{E}[x_{G,i}]$ ; then  $\mathbb{E}[b_G] \leq 4 \exp(-\mu_G)$ .

By a union bound, the probability that there exists  $G \in \mathcal{G}$  such that  $\mu_G > 2 \ln s$  and  $b_G = 1$  is at most

$$\sum_{G: \mu_G > 2 \ln s} 4 \exp(-\mu_G) < 4s \exp(-2 \ln s) = \exp(-\Omega(\log s)) \leq \exp(-\Omega(n2^{-d}/\log s)).$$

Therefore it suffices to prove that for some  $a \in \{0, 1\}$ , the targets of  $|\varphi\rangle := D(L \otimes \bigotimes_{G: \mu_G \leq 2 \ln s} G \otimes I)|0 \dots 0\rangle$  measure to  $a^n$  with probability at most  $\exp(-\Omega(n2^{-d}/\log s))$ . Henceforth we will never refer to any gate  $G$  for which  $\mu_G > 2 \ln s$ ; phrases such as “for all  $G$ ” and “ $(\cdot)_G$ ” will implicitly quantify over only those gates  $G$  for which  $\mu_G \leq 2 \ln s$ .

Let  $b = (b_G)_G$  and  $x = (x_{G,i})_{G,i}$ . Call  $x$  “good” if  $\sum_i x_{G,i} \leq c \ln s$  for all  $G$ , where  $c > 2$  is a universal constant large enough so that  $e(2e/c)^c < 1$ . A well-known Chernoff bound states that if  $S$  is a sum of independent Bernoulli random variables, and  $\mu = \mathbb{E}[S]$ , then  $P(S > t) < (e\mu/t)^t e^{-\mu}$  for all  $t > \mu$ . Therefore, by a union bound and the fact that  $\max_G \mu_G \leq 2 \ln s$ , the probability that  $x$  fails to be good is at most

$$\sum_G (e\mu_G/c \ln s)^{c \ln s} \leq s(2e/c)^{c \ln s} = (e(2e/c)^c)^{\ln s} = e^{-\Omega(\log s)} \leq \exp(-\Omega(n2^{-d}/\log s)).$$

Let  $y$  be a string of independent Bernoulli random variables distributed identically to a standard-basis measurement of  $L|0 \dots 0\rangle$ . Call the targets of  $D|y, (b_G \wedge x_{G,i})_{G,i}, 0 \dots 0\rangle$  the “output bits”, and note that they are distributed identically to a standard-basis measurement of the targets of  $|\varphi\rangle$ . If  $b$  is fixed then the output bits are a read- $2^d$  family (as functions of the independent Bernoulli random variables in  $x$  and  $y$ ). Alternatively, if  $x$  and  $y$  are fixed and  $x$  is good then the output bits are a read- $O(2^d \log s)$  family (as functions of the independent Bernoulli random variables in  $b$ ).

The rest of the proof is given in the full paper, and involves the triangle inequality. ◀

## 5 Lower Bounds for General QAC Circuits

In Section 5.1 we prove a generalization of Theorem 1.5. The proof uses the following claim, which is proved in Section 5.2 (and is obtained as a corollary of a stronger result):

► **Corollary 5.1.** *For all  $d \geq 1$ , orthogonal projections  $Q_1, \dots, Q_d$ , and states  $|\phi\rangle$ ,*

$$\|Q_d \cdots Q_1 |\phi\rangle\| \leq \exp\left(-\frac{\langle \phi | (I - Q_d) | \phi \rangle}{2d}\right).$$

Then, using this generalization of Theorem 1.5, in Sections 5.3 and 5.4 we prove Theorem 1.7(iii).

### 5.1 Proof of Theorem 1.5

Theorem 1.5 is the case of the following in which  $\mathcal{H}_1, \dots, \mathcal{H}_n$  are single-qubit Hilbert spaces,  $|\phi\rangle$  is the all-zeros state,  $Q_j = |0\rangle\langle 0|$  for all  $j$ , and  $|\psi\rangle$  is an  $n$ -nekomata.

► **Theorem 5.2.** *There is a universal constant  $c > 0$  such that the following holds. Let  $\mathcal{H}_1, \dots, \mathcal{H}_n$  be Hilbert spaces, let  $\mathcal{H}_T = \bigotimes_{j=1}^n \mathcal{H}_j$  (for “targets”), and let  $\mathcal{H}_A$  be a Hilbert space (for “ancillae”). Let  $|\phi\rangle = |\phi_1, \dots, \phi_n, \phi_A\rangle$  for some states  $|\phi_j\rangle \in \mathcal{H}_j, j \in [n] \cup \{A\}$ . Let  $Q_j$  be an orthogonal projection on  $\mathcal{H}_j$  for  $j \in [n]$ , and let  $|\psi\rangle$  be a state in  $\mathcal{H}_T \otimes \mathcal{H}_A$  that measures to  $\bigotimes_{j=1}^n Q_j \otimes I_{\mathcal{H}_A}$  and to  $\bigotimes_{j=1}^n (I - Q_j) \otimes I_{\mathcal{H}_A}$  each with probability  $1/2$ . Let  $C$  be a depth- $d$  QAC circuit on  $\mathcal{H}_T \otimes \mathcal{H}_A$  with at most  $cn/(d+1)$  multi-qubit gates acting on  $\mathcal{H}_T$ . Then,  $|\langle \psi | C | \phi \rangle|^2 \leq 1/2 + \exp(-\Omega(n/(d+1)))$ .*

**Abridged proof.** By Proposition 2.1 we may write  $C = DL$  for some layer of single-qubit gates  $L$  and QAC circuit  $D$ , where  $D$  has the same topology as  $C$  and consists only of multi-qubit  $R_\otimes$  gates. Since  $L|\phi\rangle$  factors as a product state in the same way that  $|\phi\rangle$  does, we may assume without loss of generality that  $C$  consists only of multi-qubit  $R_\otimes$  gates, by replacing  $C$  and  $|\phi\rangle$  with  $D$  and  $L|\phi\rangle$  respectively.

We now generalize Lemma 4.4 from nekomata to states such as  $|\psi\rangle$ . Let  $Q = \bigotimes_{j=1}^n Q_j \otimes I_{\mathcal{H}_A}$  and  $Q' = \bigotimes_{j=1}^n (I - Q_j) \otimes I_{\mathcal{H}_A}$ , and let  $|\varphi\rangle = C|\phi\rangle$ . Since  $|\psi\rangle$  measures to  $Q + Q'$  with probability 1, it follows from the triangle inequality and Cauchy-Schwarz that

$$\begin{aligned} |\langle\psi|\varphi\rangle|^2 &= |\langle\psi|(Q + Q')|\varphi\rangle|^2 \leq (|\langle\psi|Q|\varphi\rangle| + |\langle\psi|Q'|\varphi\rangle|)^2 \\ &\leq (\|Q|\varphi\rangle\| \cdot \|Q|\psi\rangle\| + \|Q'|\varphi\rangle\| \cdot \|Q'|\psi\rangle\|)^2 = (\|Q|\varphi\rangle\|/\sqrt{2} + \|Q'|\varphi\rangle\|/\sqrt{2})^2 \\ &= \langle\varphi|(Q + Q')|\varphi\rangle/2 + \|Q|\varphi\rangle\| \cdot \|Q'|\varphi\rangle\| \leq 1/2 + \min(\|Q|\varphi\rangle\|, \|Q'|\varphi\rangle\|), \end{aligned}$$

so it suffices to prove that  $\min(\|Q|\varphi\rangle\|, \|Q'|\varphi\rangle\|) \leq \exp(-\Omega(n/(d+1)))$ .

Since  $\sum_{j=1}^n \langle\phi_j|Q_j|\phi_j\rangle + \sum_{j=1}^n \langle\phi_j|(I - Q_j)|\phi_j\rangle = n$ , either  $\sum_{j=1}^n \langle\phi_j|Q_j|\phi_j\rangle \geq n/2$  or  $\sum_{j=1}^n \langle\phi_j|(I - Q_j)|\phi_j\rangle \geq n/2$ . Assume without loss of generality that  $\sum_{j=1}^n \langle\phi_j|(I - Q_j)|\phi_j\rangle \geq n/2$ . We will prove that  $\|Q|\varphi\rangle\| \leq \exp(-\Omega(n/(d+1)))$ .

Let  $\mathcal{G}$  be the set of gates in  $C$ , ordered such that  $C = \prod_{G \in \mathcal{G}} G$  (where each gate  $G$  is implicitly tensored with the identity). Also let  $\mathcal{G}_T \subseteq \mathcal{G}$  be the set of gates in  $C$  that act on  $\mathcal{H}_T$ . For  $G \in \mathcal{G}_T$  let  $|\theta_G\rangle$  be the mono-product state, specified up to a phase factor, such that  $G = R_{\theta_G} = I - 2|\theta_G\rangle\langle\theta_G|$ . Let  $F$  be the set of functions with domain  $\mathcal{G}$  that map each gate  $G$  in  $\mathcal{G}_T$  to either  $I$  or  $|\theta_G\rangle\langle\theta_G|$ , and map each gate  $G$  in  $\mathcal{G} \setminus \mathcal{G}_T$  to  $G$  itself. Then  $C = \sum_{f \in F} (-2)^{|\{G: f(G)=|\theta_G\rangle\langle\theta_G|\}|} \prod_{G \in \mathcal{G}} f(G)$ , so by the triangle inequality,

$$\|Q|\varphi\rangle\| = \|QC|\phi\rangle\| \leq \sum_{f \in F} 2^{|\{G: f(G)=|\theta_G\rangle\langle\theta_G|\}|} \cdot \max_{f \in F} \left\| Q \prod_{G \in \mathcal{G}} f(G) \cdot |\phi\rangle \right\|.$$

By assumption,  $|\mathcal{G}_T| \leq cn/(d+1)$  (for a constant  $c$  to be specified later), so

$$\sum_{f \in F} 2^{|\{G: f(G)=|\theta_G\rangle\langle\theta_G|\}|} = \sum_{S \subseteq \mathcal{G}_T} 2^{|S|} = \prod_{G \in \mathcal{G}_T} (2^0 + 2^1) = 3^{|\mathcal{G}_T|} \leq 3^{cn/(d+1)}.$$

Consider an arbitrary function  $f \in F$ . In the full paper we write  $\|Q \prod_{G \in \mathcal{G}} f(G) \cdot |\phi\rangle\|$  as a product of  $n+1$  terms, one of which we bound by 1, and the other  $n$  of which we bound individually using Corollary 5.1. The result is that

$$\left\| Q \prod_{G \in \mathcal{G}} f(G) \cdot |\phi\rangle \right\| \leq \prod_{j=1}^n \exp\left(-\frac{\langle\phi_j|(I - Q_j)|\phi_j\rangle}{2(d+1)}\right) \leq \exp\left(-\frac{n/2}{2(d+1)}\right).$$

Altogether this implies that  $\|Q|\varphi\rangle\| \leq \exp((c \ln 3 - 1/4) \cdot n/(d+1))$ , and the result follows by taking  $c < 1/(4 \ln 3)$ .  $\blacktriangleleft$

## 5.2 Proof of Corollary 5.1

Let  $\Delta(|\alpha\rangle, |\beta\rangle) = \arccos |\langle\alpha|\beta\rangle|$ ; we will abbreviate this as  $\Delta(\alpha, \beta)$ . In the full paper we prove the following:

► **Lemma 5.3.** *The function  $\Delta$  satisfies the triangle inequality, i.e.  $\Delta(\alpha, \gamma) \leq \Delta(\alpha, \beta) + \Delta(\beta, \gamma)$  for all states  $|\alpha\rangle, |\beta\rangle, |\gamma\rangle$ .*

► **Remark.** For intuition as to why Lemma 5.3 is true, consider the similarly defined function  $\Delta'(u, v) = \arccos\langle u, v \rangle$  for unit vectors  $u, v \in \mathbb{R}^3$ , where  $\langle \cdot, \cdot \rangle$  denotes the usual inner product on  $\mathbb{R}^3$ . It is well known that  $\Delta'(u, v)$  equals the angle between  $u$  and  $v$ , which equals the length of the arc (Figure 5) formed by traversing a great circle on the unit sphere from  $u$  to  $v$  in the shorter of the two directions. This arc is known to be the shortest path on the unit sphere between  $u$  and  $v$ , so  $\Delta'$  represents distance on the unit sphere. See the full paper for more related discussion.

► **Proposition 5.4.** For all  $d \geq 1$ , nonzero orthogonal projections  $Q_d$ , and states  $|\phi\rangle$ ,

$$\max_{Q_1, \dots, Q_{d-1}} \|Q_d Q_{d-1} \cdots Q_1 |\phi\rangle\| = \cos\left(\frac{\arccos \|Q_d |\phi\rangle\|}{d}\right)^d,$$

where the maximum is taken over all orthogonal projections  $Q_1, \dots, Q_{d-1}$ .

**Abridged proof.** We first prove an analogous statement about rank-1 orthogonal projections, specifically that for all states  $|\theta_0\rangle$  and  $|\theta_d\rangle$ ,

$$\max_{|\theta_1\rangle, \dots, |\theta_{d-1}\rangle} \left| \prod_{j=1}^d \langle \theta_{j-1} | \theta_j \rangle \right| = \cos\left(\frac{\arccos |\langle \theta_0 | \theta_d \rangle|}{d}\right)^d. \quad (3)$$

Then, in the full paper, we prove that the original proposition follows from this rank-1 analogue.

On the image of  $\Delta$ , i.e. on the interval  $[0, \pi/2]$ , the cosine function is decreasing and concave. Therefore for all states  $|\theta_1\rangle, \dots, |\theta_{d-1}\rangle$ , by the AM-GM inequality, Jensen's inequality, and Lemma 5.3,

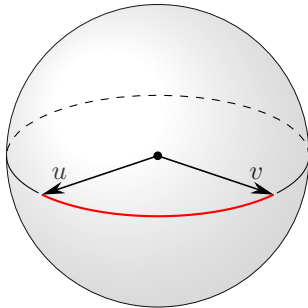
$$\begin{aligned} \left| \prod_{j=1}^d \langle \theta_{j-1} | \theta_j \rangle \right|^{1/d} &\leq \frac{1}{d} \sum_{j=1}^d |\langle \theta_{j-1} | \theta_j \rangle| = \frac{1}{d} \sum_{j=1}^d \cos \Delta(\theta_{j-1}, \theta_j) \leq \cos\left(\frac{1}{d} \sum_{j=1}^d \Delta(\theta_{j-1}, \theta_j)\right) \\ &\leq \cos\left(\frac{\Delta(\theta_0, \theta_d)}{d}\right) = \cos\left(\frac{\arccos |\langle \theta_0 | \theta_d \rangle|}{d}\right). \end{aligned}$$

In the full paper we give an example (Figure 6) which shows that this bound is tight. ◀

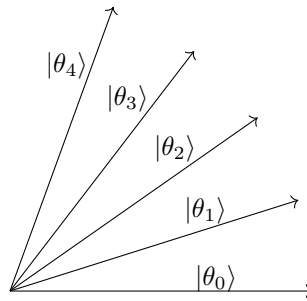
► **Reminder (Corollary 5.1).** For all  $d \geq 1$ , orthogonal projections  $Q_1, \dots, Q_d$ , and states  $|\phi\rangle$ ,

$$\|Q_d \cdots Q_1 |\phi\rangle\| \leq \exp\left(-\frac{\langle \phi | (I - Q_d) | \phi \rangle}{2d}\right).$$

**Proof sketch.** The case  $Q_d = 0$  is trivial. The case  $Q_d \neq 0$  is handled using Proposition 5.4 and the Lagrange remainder theorem. ◀



■ **Figure 5** A geodesic on the sphere.



■ **Figure 6** An optimal choice of  $|\theta_1\rangle, \dots, |\theta_{d-1}\rangle$  in the  $d = 4$  case of Equation (3).

### 5.3 Simplifying Depth-2 QAC Circuits by Measuring Ancillae

For a one-qubit state  $|\psi\rangle$ , let the  $|\psi\rangle$  basis be an orthonormal basis of  $\mathbb{C}^2$  that includes  $|\psi\rangle$ . (We refer to “the”  $|\psi\rangle$  basis because, up to a phase factor, there is a unique state orthogonal to  $|\psi\rangle$ .)

► **Lemma 5.5.** *Let  $\mathcal{H}_1$  be a one-qubit Hilbert space, and let  $\mathcal{H}_2$  and  $\mathcal{H}_3$  be Hilbert spaces on arbitrary numbers of qubits. Then for all  $|\psi\rangle \in \mathcal{H}_1$ ,  $|\theta\rangle \in \mathcal{H}_2$ ,  $|\phi\rangle \in \mathcal{H}_1 \otimes \mathcal{H}_2 \otimes \mathcal{H}_3$ , the following two procedures generate identically distributed random states in  $\mathcal{H}_1 \otimes \mathcal{H}_2 \otimes \mathcal{H}_3$ :*

- *measure the  $\mathcal{H}_1$  qubit of  $(R_{|\psi,\theta\rangle} \otimes I_{\mathcal{H}_3})|\phi\rangle$  in the  $|\psi\rangle$  basis;*
- *measure the  $\mathcal{H}_1$  qubit of  $|\phi\rangle$  in the  $|\psi\rangle$  basis, and then, conditioned on the outcome being  $|\psi\rangle$ , apply  $R_\theta$  on  $\mathcal{H}_2$ .*

**Proof.** This follows easily from the fact that  $R_{|\psi,\theta\rangle} = (I - |\psi\rangle\langle\psi|) \otimes I + |\psi\rangle\langle\psi| \otimes R_\theta$ . ◀

Theorem 1.7(iii) is clearly equivalent to the statement that if  $C$  is a depth-2 QAC circuit, then any  $n$  designated “target” qubits of  $C|0 \dots 0\rangle$  measure to  $|\otimes_n\rangle$  with probability at most  $1/2 + \exp(-\Omega(n))$ . The following is the starting point for our proof:

► **Proposition 5.6.** *Let  $p$  and  $|\psi\rangle$  be such that for some depth-2 QAC circuit  $C$ , designated “target” qubits of  $C|0 \dots 0\rangle$  measure to  $|\psi\rangle$  with probability  $p$ . Then there exist layers of  $R_\otimes$  gates  $L_2, L_1$  and a mono-product state  $|\phi\rangle$  such that for some partition of the qubits of  $L_2 L_1 |\phi\rangle$  into “targets” and “ancillae”,*

- (i) *the targets of  $L_2 L_1 |\phi\rangle$  measure to  $|\psi\rangle$  with probability at least  $p$ ;*
- (ii) *for all  $k \in \{1, 2\}$ , every ancilla is acted on by a gate in  $L_k$ , and every gate in  $L_k$  acts on at least one target.*

► **Remark.** Although not necessary for our purposes, using Proposition 2.1 it is easy to generalize the following argument to show that the gates in  $L_2$  and  $L_1$  may be assumed to be multi-qubit gates.

**Proof sketch.** By Proposition 2.1 there exist  $L_2, L_1, |\phi\rangle$  satisfying (i) but not necessarily (ii). We measure selected ancillae in an appropriate product basis; doing so simplifies the circuit due to Lemma 5.5, and in expectation does not change the probability that the targets measure to  $|\psi\rangle$ . ◀

### 5.4 Proof of Theorem 1.7(iii)

The  $\delta = 1$  case of the following is Markov’s inequality:

► **Lemma 5.7.** *Let  $0 < \delta \leq 1$ , let  $a > 0$ , and let  $X$  be a nonnegative random variable. Then there exists  $t \in [a, ae^{\delta^{-1}-1}]$  such that  $P(X \geq t) \leq \delta \mathbb{E}[X]/t$ .*

► **Remark.** The intuition behind our use of Lemma 5.7 is as follows. Theorem 5.2 implies that depth-2 QAC circuits require size at least  $\Omega(n)$  to approximately construct  $|\otimes_n, \psi\rangle$ , and Proposition 5.6 implies that depth-2 QAC circuits that approximately construct  $|\otimes_n, \psi\rangle$  have size at most  $2n$  without loss of generality, so these bounds are “just a constant factor” away from implying that depth-2 QAC circuits of arbitrary size cannot approximately construct  $|\otimes_n, \psi\rangle$ . This is analogous to how Markov’s inequality is “just a factor of  $\delta$ ” away from the conclusion of Lemma 5.7.

**Proof sketch.** Assume the contrary, write  $\mathbb{E}[X] = \int_0^\infty P(X \geq t) dt$ , and obtain the contradiction  $\mathbb{E}[X] > \mathbb{E}[X]$  using the assumed lower bound on  $P(X \geq t)$ . ◀

► **Theorem 5.8** (Turán’s theorem<sup>7</sup>). *Let  $\mathcal{G}$  be a simple undirected graph on  $n$  vertices, and let  $d$  be the average degree of the vertices in  $\mathcal{G}$ . Then  $\mathcal{G}$  contains an independent set of size at least  $n/(d+1)$ .*

In the full paper, we repeat the proof of Theorem 5.8 exposited by Alon and Spencer [1].

► **Remark.** For the intuition behind our use of Theorem 5.8, recall the discussion of disjoint light cones from Section 1.2.7.

Recall that  $|\phi\rangle, C, |\psi\rangle, (Q_j)_j$  are variables from the statement of Theorem 5.2. In upcoming applications of Theorem 5.2 we will refer to  $|\phi\rangle$  as the “input state”,  $C$  as the “circuit”,  $|\psi\rangle$  as the “desired output state”, and  $(Q_j)_j$  as “projections”.

► **Remark.** We will not actually use the full strength of Theorem 5.2, in the sense that we will always upper-bound the number of multi-qubit gates acting on the targets by upper-bounding the *total* number of gates. One could instead use the full strength of Theorem 5.2 in this regard, and forgo the use of Proposition 5.6 entirely by measuring selected ancillae all at once later in the proof, but we consider the current presentation to be simpler.

► **Reminder** (Theorem 1.7(iii), paraphrased). *If  $C$  is a depth-2 QAC circuit, then any  $n$  designated “target” qubits of  $C|0\dots 0\rangle$  measure to  $|\boxtimes_n\rangle$  with probability at most  $1/2 + \exp(-\Omega(n))$ .*

**Abridged proof.** Let  $L_2, L_1$  be layers of  $R_\otimes$  gates and let  $|\phi\rangle$  be a mono-product state, with  $n$  qubits designated as targets and all other qubits designated as ancillae. Assume that for all  $k \in \{1, 2\}$ , every ancilla is acted on by a gate in  $L_k$ , and every gate in  $L_k$  acts on at least one target. By Proposition 5.6 it suffices to prove that the targets of  $L_2 L_1 |\phi\rangle$  measure to  $|\boxtimes_n\rangle$  with probability at most  $1/2 + \exp(-\Omega(n))$ .

Let  $c$  be the constant from Theorem 5.2, and let  $\gamma = (c/2)(c/3)/(1+c/2)$  and  $\delta = (c/2)\gamma^2$ . Since Theorem 5.2 remains true if  $c$  is replaced by any constant between 0 and  $c$ , we may take  $c$  to be small enough so that  $\gamma, \delta \leq 1$ .

For a circuit  $C$  let  $|C|$  denote the number of gates in  $C$ , and write “ $G \in C$ ” to denote that  $G$  is a gate in  $C$ . First consider the case where  $|L_2| \leq \gamma n$ . It suffices to prove that  $L_2 L_1 |\phi\rangle$  and  $|\boxtimes_n, \psi\rangle$  have fidelity at most  $1/2 + \exp(-\Omega(n))$  for all states  $|\psi\rangle$ . If  $|L_1| \leq n(c/3)/(1+c/2)$  then  $|L_1| + |L_2| \leq (c/3)n$ , and the result follows from applying Theorem 5.2 with input state  $|\phi\rangle$ , circuit  $L_2 L_1$ , desired output state  $|\boxtimes_n, \psi\rangle$ , and  $n$  one-qubit projections  $|0\rangle\langle 0|$  acting on the targets. Alternatively, if  $|L_1| \geq n(c/3)/(1+c/2)$  then  $|L_2| \leq (c/2)|L_1|$ , and the result follows from applying Theorem 5.2 with input state  $L_1 |\phi\rangle$ , circuit  $L_2$ , desired output state  $|\boxtimes_n, \psi\rangle$ , and for every gate  $G \in L_1$  the projection  $|0\rangle\langle 0| \otimes I$  on the support of  $G$ , where  $|0\rangle\langle 0|$  acts on one of the targets acted on by  $G$ . (Here we used the fact that  $1/2 + \exp(-\Omega(|L_1|)) \leq 1/2 + \exp(-\Omega(n))$  by our assumption about  $|L_1|$ .)

The rest of the proof, i.e. the analysis of the case where  $|L_2| \geq \gamma n$ , is given in the full paper, and is a (relatively complicated) application of previously discussed ideas. ◀

<sup>7</sup> Usually Turán’s theorem is phrased as saying that dense graphs have large cliques, whereas Theorem 5.8 says that sparse graphs have large independent sets. These statements are equivalent, because taking the complement of a graph turns cliques into independent sets and vice versa.

## References

- 1 Noga Alon and Joel Spencer. *The Probabilistic Method*. Wiley Series in Discrete Mathematics and Optimization. John Wiley & Sons, 4 edition, 2016.
- 2 Adam Bene Watts, Robin Kothari, Luke Schaeffer, and Avishay Tal. Exponential separation between shallow quantum circuits and unbounded fan-in shallow classical circuits. In *Symposium on the Theory of Computing*, pages 515–526, 2019. doi:10.1145/3313276.3316404.
- 3 Debajyoti Bera. A lower bound method for quantum circuits. *Information Processing Letters*, 111(15):723–726, 2011. doi:10.1016/j.ipl.2011.05.002.
- 4 Debajyoti Bera, Frederic Green, and Steven Homer. Small depth quantum circuits. *ACM SIGACT News*, 38(2):35–50, 2007. doi:10.1145/1272729.1272739.
- 5 Maosen Fang, Stephen Fenner, Frederic Green, Steven Homer, and Yong Zhang. Quantum lower bounds for fanout. *Quantum Information & Computation*, 6(1):46–57, 2006. arXiv:quant-ph/0312208.
- 6 Dmitry Gavinsky, Shachar Lovett, Michael Saks, and Srikanth Srinivasan. A tail bound for read- $k$  families of functions. *Random Structures & Algorithms*, 47(1):99–108, 2015. doi:10.1002/rsa.20532.
- 7 Pranav Gokhale, Samantha Koretsky, Shilin Huang, Swarnadeep Majumder, Andrew Drucker, Kenneth R. Brown, and Frederic T. Chong. Quantum fan-out: circuit optimizations and technology modeling. arXiv, 2020. arXiv:2007.04246.
- 8 Frederic Green, Steven Homer, Cristopher Moore, and Christopher Pollett. Counting, fanout, and the complexity of quantum ACC. *Quantum Information & Computation*, 2(1):35–65, 2002. arXiv:quant-ph/0106017.
- 9 Andrew Y. Guo, Abhinav Deshpande, Su-Kuan Chu, Zachary Eldredge, Przemyslaw Bienias, Dhruv Devulapalli, Yuan Su, Andrew M. Childs, and Alexey V. Gorshkov. Implementing a fast unbounded quantum fanout gate using power-law interactions. arXiv, 2020. arXiv:2007.00662.
- 10 Johan Håstad. Almost optimal lower bounds for small depth circuits. In *Symposium on the Theory of Computing*, pages 6–20, 1986. doi:10.1145/12130.12132.
- 11 Peter Høyer and Robert Špalek. Quantum fan-out is powerful. *Theory of Computing*, 1(5):81–103, 2005. doi:10.4086/toc.2005.v001a005.
- 12 Alastair Kay. *Tutorial on the Quantikz package*, 2020. doi:10.17637/rh.7000520.
- 13 Daniel Padé, Stephen Fenner, Daniel Grier, and Thomas Thierauf. Depth-2 QAC circuits cannot simulate quantum parity. arXiv, 2020. arXiv:2005.12169.
- 14 Yasuhiro Takahashi and Seiichiro Tani. Collapse of the hierarchy of constant-depth exact quantum circuits. *Computational Complexity*, 25(4):849–881, 2016. doi:10.1007/s00037-016-0140-0.



# Query Complexity Lower Bounds for Local List-Decoding and Hard-Core Predicates (Even for Small Rate and Huge Lists)

Noga Ron-Zewi

University of Haifa, Israel  
noga@cs.haifa.ac.il

Ronen Shaltiel

University of Haifa, Israel  
ronen@cs.haifa.ac.il

Nithin Varma

University of Haifa, Israel  
nvarma@bu.edu

---

## Abstract

A binary code  $\text{Enc} : \{0, 1\}^k \rightarrow \{0, 1\}^n$  is  $(\frac{1}{2} - \epsilon, L)$ -list decodable if for every  $w \in \{0, 1\}^n$ , there exists a set  $\text{List}(w)$  of size at most  $L$ , containing all messages  $m \in \{0, 1\}^k$  such that the relative Hamming distance between  $\text{Enc}(m)$  and  $w$  is at most  $\frac{1}{2} - \epsilon$ . A  $q$ -query local list-decoder for  $\text{Enc}$  is a randomized procedure  $\text{Dec}$  that when given oracle access to a string  $w$ , makes at most  $q$  oracle calls, and for every message  $m \in \text{List}(w)$ , with high probability, there exists  $j \in [L]$  such that for every  $i \in [k]$ , with high probability,  $\text{Dec}^w(i, j) = m_i$ .

We prove lower bounds on  $q$ , that apply even if  $L$  is huge (say  $L = 2^{k^{0.9}}$ ) and the rate of  $\text{Enc}$  is small (meaning that  $n \geq 2^k$ ):

- For  $\epsilon = 1/k^\nu$  for some constant  $0 < \nu < 1$ , we prove a lower bound of  $q = \Omega(\frac{\log(1/\delta)}{\epsilon^2})$ , where  $\delta$  is the error probability of the local list-decoder. This bound is tight as there is a matching upper bound by Goldreich and Levin (STOC 1989) of  $q = O(\frac{\log(1/\delta)}{\epsilon^2})$  for the Hadamard code (which has  $n = 2^k$ ). This bound extends an earlier work of Grinberg, Shaltiel and Viola (FOCS 2018) which only works if  $n \leq 2^{k^\nu}$  and the number of coins tossed by  $\text{Dec}$  is small (and therefore does not apply to the Hadamard code, or other codes with low rate).
- For smaller  $\epsilon$ , we prove a lower bound of roughly  $q = \Omega(\frac{1}{\sqrt{\epsilon}})$ . To the best of our knowledge, this is the first lower bound on the number of queries of local list-decoders that gives  $q \geq k$  for small  $\epsilon$ .

Local list-decoders with small  $\epsilon$  form the key component in the celebrated theorem of Goldreich and Levin that extracts a *hard-core predicate* from a one-way function. We show that black-box proofs *cannot* improve the Goldreich-Levin theorem and produce a hard-core predicate that is hard to predict with probability  $\frac{1}{2} + \frac{1}{\ell^{\omega(1)}}$  when provided with a one-way function  $f : \{0, 1\}^\ell \rightarrow \{0, 1\}^\ell$ , where  $f$  is such that circuits of size  $\text{poly}(\ell)$  cannot invert  $f$  with probability  $\rho = 1/2^{\sqrt{\ell}}$  (or even  $\rho = 1/2^{\Omega(\ell)}$ ). This limitation applies to any proof by black-box reduction (even if the reduction is allowed to use nonuniformity and has oracle access to  $f$ ).

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Computational complexity and cryptography

**Keywords and phrases** Local list-decoding, Hard-core predicates, Black-box reduction, Hadamard code

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.33

**Related Version** A full version of the paper is available at <https://eccc.weizmann.ac.il/report/2020/133/>.



© Noga Ron-Zewi, Ronen Shaltiel, and Nithin Varma;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 33; pp. 33:1–33:18



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

**Funding** Noga Ron-Zewi: Partially supported by ISF grant 735/20.

Ronen Shaltiel: Partially supported by ISF grant 1628/17.

Nithin Varma: Partially supported by ISF grant 497/17, and Israel PBC Fellowship for Outstanding Postdoctoral Researchers from India and China.

**Acknowledgements** We are grateful to Ilan Newman for participating in early stages of this research and for many helpful discussions.

## 1 Introduction

We prove limitations on local list-decoding algorithms and on reductions establishing hard-core predicates.

### 1.1 Locally list-decodable codes

List-decodable codes are a natural extension of (uniquely decodable) error-correcting codes, as it allows (list) decoding for error regimes where unique decoding is impossible. This is an extensively studied area; see [8] for a survey. In this paper, we will be interested in list-decoding of binary codes.

► **Definition 1** (List-decodable code). *For a function  $\text{Enc} : \{0, 1\}^k \rightarrow \{0, 1\}^n$ , and  $w \in \{0, 1\}^n$ , we define*

$$\text{List}_\alpha^{\text{Enc}}(w) = \{m \in \{0, 1\}^k : \text{dist}(\text{Enc}(m), w) \leq \alpha\}.$$
<sup>1</sup>

*We say that  $\text{Enc}$  is  $(\alpha, L)$ -list-decodable if for every  $w \in \{0, 1\}^n$ ,  $|\text{List}_\alpha^{\text{Enc}}(w)| \leq L$ .*

The task of *algorithmic* list-decoding is to produce the list  $\text{List}_\alpha^{\text{Enc}}(w)$  on input  $w \in \{0, 1\}^n$ .

*Local* unique decoding algorithms are algorithms that given an index  $i \in [k]$ , make few oracle queries to  $w$ , and reproduce the bit  $m_i$  (with high probability over the choice of their random coins), where  $m$  denotes the unique codeword close to  $w$ . This notion of *local decoding* has many connections and applications in computer science and mathematics [18].

We will be interested in *local* list-decoding algorithms that receive oracle access to a received word  $w \in \{0, 1\}^n$ , as well as inputs  $i \in [k]$  and  $j \in [L]$ . We will require that for every  $m \in \text{List}_\alpha^{\text{Enc}}(w)$ , with high probability, there exists a  $j \in [L]$  such that for every  $i \in [k]$ , when Dec receives oracle access to  $w$  and inputs  $i, j$ , it produces  $m_i$  with high probability over its choice of random coins. This motivates the next definition.

► **Definition 2** (Randomized local computation). *We say that a procedure  $P(i, r)$  locally computes a string  $m \in \{0, 1\}^k$  with error  $\delta$ , if for every  $i \in [k]$ ,  $\Pr[P(i, R) = m_i] \geq 1 - \delta$  (where the probability is over a uniform choice of the “string of random coins”  $R$ ).*

The definition of local list-decoders considers an algorithmic scenario that works in two steps:

- At the first step (which can be thought of as a preprocessing step) the local list-decoder Dec is given oracle access to  $w$  and an index  $j \in [L]$ . It tosses random coins (which we denote by  $r^{\text{shared}}$ ).
- At the second step, the decoder receives the additional index  $i \in [k]$ , and tosses additional coins  $r$ .

<sup>1</sup> For two strings  $x, y \in \{0, 1\}^n$  we use  $\text{dist}(x, y)$  to denote the relative Hamming distance between  $x$  and  $y$ , namely,  $\text{dist}(x, y) = |\{i \in [n] : x_i \neq y_i\}|/n$ .

- It is required that for every  $w \in \{0, 1\}^n$  and  $m \in \text{List}_\alpha^{\text{Enc}}(w)$ , with probability  $2/3$  over the choice of the shared coins  $r^{\text{shared}}$ , there exists  $j \in [L]$  such that when the local list-decoder receives  $j$ , it locally computes  $m$  (using its “non-shared” coins  $r$ ). The definition uses two types of random coins because the coins  $r^{\text{shared}}$  are “shared” between different choices of  $i \in [k]$  and allow different  $i$ ’s to “coordinate”. The coins  $r$ , are chosen independently for different choices of  $i \in [k]$ .

This is formally stated in the next definition.

► **Definition 3** (Local list-decoder). *Let  $\text{Enc} : \{0, 1\}^k \rightarrow \{0, 1\}^n$  be a function. An  $(\alpha, L, q, \delta)$ -local list-decoder (LLD) for  $\text{Enc}$  is an oracle procedure  $\text{Dec}^{(\cdot)}$  that receives oracle access to a word  $w \in \{0, 1\}^n$ , and makes at most  $q$  calls to the oracle. The procedure  $\text{Dec}$  also receives inputs:*

- $i \in [k]$  : The index of the symbol that it needs to decode.
- $j \in [L]$  : An index to the list.
- Two strings  $r^{\text{shared}}, r$  that are used as random coins.

*It is required that for every  $w \in \{0, 1\}^n$ , and for every  $m \in \text{List}_\alpha^{\text{Enc}}(w)$ , with probability at least  $2/3$  over choosing a uniform string  $r^{\text{shared}}$ , there exists  $j \in [L]$  such that the procedure*

$$P_{w,j,r^{\text{shared}}}(i, r) = \text{Dec}^w(i, j, r^{\text{shared}}, r)$$

*locally computes  $m$  with error  $\delta$ . If we omit  $\delta$ , then we mean  $\delta = 1/3$ .*

► **Remark 4** (On the generality of Definition 3). The goal of this paper is to prove lower bounds on local list-decoders, and so, making local list-decoders as general as possible, makes our results stronger. We now comment on the generality of Definition 3.

- In Definition 3 we do not require that  $L = |\text{List}_\alpha^{\text{Enc}}(w)|$ , and allow the local list-decoder to use a larger  $L$ . This means that on a given  $w$ , there may be many choices of  $j \in [L]$  such that the procedure  $P_{w,j,r^{\text{shared}}}(i, r) = \text{Dec}^w(i, j, r^{\text{shared}}, r)$  locally computes messages  $m \notin \text{List}_\alpha^{\text{Enc}}(w)$ .
- In Definition 3 we do not place any restriction on the number of random coins used by the local list-decoder, making the task of local list-decoding easier.
- We allow  $\text{Dec}$  to make *adaptive* queries to its oracle.
- We are only interested in the total number of queries made by the combination of the two steps. It should be noted that w.l.o.g., a local list-decoder can defer all its queries to the second step (namely, after it receives the input  $i$ ), and so, this definition captures local list-decoding algorithms which make queries to the oracle at both steps.
- To the best of our knowledge, all known local list-decoders in the literature are of the form presented in Definition 3.

### 1.1.1 Lower bounds on the query complexity of local list-decoders

In this paper we prove lower bounds on the number of queries  $q$  of  $(\frac{1}{2} - \epsilon, L, q, \delta)$ -local list-decoders. Our goal is to show that the number of queries  $q$  has to be large, when  $\epsilon$  is *small*. Our lower bounds apply even if the size of the list  $L$  is huge and approaches  $2^k$  (note that a local list-decoder can trivially achieve  $L = 2^k$  with a list of all messages). Our lower bounds also apply even if the rate of the code is very small, and  $n \geq 2^k$ .

We remark that this parameter regime is very different than the one studied in lower bounds on the number of queries of local decoders for *uniquely* decodable codes (that is, for  $L = 1$ ). By the Plotkin bound, uniquely decodable codes cannot have  $\epsilon < \frac{1}{4}$ , and so, the main focus in uniquely decodable codes is to show that local decoders for codes with “good rate” and “large”  $\epsilon = \Omega(1)$ , must make many queries. In contrast, we are interested in the case where  $\epsilon$  is small, and want to prove lower bounds that apply to huge lists and small rate.

### Lower bounds for large $\epsilon$

Our first result is a tight lower bound of  $q = \Omega(\frac{\log(1/\delta)}{\epsilon^2})$  on the number of queries, assuming  $\epsilon$  is sufficiently large, namely  $\epsilon \geq \frac{1}{k^\nu}$  for some constant  $0 < \nu \leq 1$ .

► **Theorem 5** (Tight lower bounds for large  $\epsilon$ ). *There exists a universal constant  $\nu > 0$  such that for any  $L \leq 2^{k^{0.9}}$ ,  $\epsilon \in (k^{-\nu}, \frac{1}{4})$ , and  $\delta \in (k^{-\nu}, \frac{1}{3})$ , we have that every  $(\frac{1}{2} - \epsilon, L, q, \delta)$ -local list-decoder for  $\text{Enc} : \{0, 1\}^k \rightarrow \{0, 1\}^n$  must have  $q = \Omega(\frac{\log(1/\delta)}{\epsilon^2})$ .*

Theorem 5 is tight in the sense that the Hadamard code (which has length  $n = 2^k$ ) has  $(\frac{1}{2} - \epsilon, L = O(1/\epsilon^2), q = O(\frac{\log(1/\delta)}{\epsilon^2}), \delta)$  local list-decoders [6]. In fact, the Hadamard code was the motivation for this research, and is a running example in this paper.

Our results show that even if we allow list sizes  $L$  which approach  $2^k$ , it is impossible to reduce the number of queries for the Hadamard code. Our results also show that even if we are willing to allow very small rate  $n \geq 2^k$ , and huge list sizes, it is impossible to have codes whose local list-decoders make fewer queries than the local list-decoders for the Hadamard code.

### Comparison to previous work

Theorem 5 improves and extends an earlier work by Grinberg, Shaltiel and Viola [7] that gave the same bound of  $q = \Omega(\frac{\log(1/\delta)}{\epsilon^2})$  for a more limited parameter regime: Specifically, in [7], for the lower bound to hold, it is also required that  $n \leq 2^{k^\nu}$ , for some constant  $\nu > 0$ , and that the total number of coins tossed by the local list-decoder is less than  $k^\nu - \log L$ .<sup>2</sup> We stress that because of these two limitations, the lower bounds of [7] do not apply to the Hadamard code and other low rate codes.

### Extensions to large alphabet and erasures

The scenario that we consider in Theorem 5 has *binary* alphabet, and decoding from *errors*. We remark that in the case of large alphabets, or decoding from erasures, there are local list-decoders which achieve  $q = O(\frac{\log(1/\delta)}{\epsilon})$  (which is smaller than what is possible for binary alphabet and decoding from errors), as was shown for the case of Hadamard codes in [11]. Our results extend to give a matching lower bound of  $q = \Omega(\frac{\log(1/\delta)}{\epsilon})$  for decoding from erasures (for any alphabet size), and also the same lower bound on decoding from errors for any alphabet size. The exact details are deferred to the final version.

### Lower bounds for small $\epsilon$

The best bound on  $q$  that Theorem 5 (as well as the aforementioned lower bounds of [7]) can give is  $q \geq k^{\Omega(1)}$ . The next theorem shows that even for small  $\epsilon < 1/k$ , we can obtain a lower bound on  $q$  which is polynomial in  $1/\epsilon$ .

► **Theorem 6** (Lower bounds for small  $\epsilon$ ). *There exist universal constants  $\beta, c_1, c_2 > 0$  such that for every  $L \leq \beta \cdot 2^k$ ,  $\delta < \frac{1}{3}$  and  $\epsilon \geq \frac{\beta}{\sqrt{n}}$  we have that every  $(\frac{1}{2} - \epsilon, L, q, \delta)$ -local list-decoder for  $\text{Enc} : \{0, 1\}^k \rightarrow \{0, 1\}^n$  must have  $q \geq \frac{1}{c_1 \log(k) \cdot \sqrt{\epsilon}} - c_2 \log L$ .*

<sup>2</sup> The work of [7] is concerned with proving lower bounds on the number of queries of “nonuniform reductions for hardness amplification” [17, 15, 3, 7]. As explained in [17, 15, 3, 7] such lower bounds translate into lower bounds on local list-decoders, by “trading” the random coins of a local list-decoder for “nonuniform advice” for the reduction, and proving a lower bound on the number of queries made by the reduction.

Note that for sufficiently small  $\epsilon = 1/(\log k)^{\omega(1)}$ , we get  $q = \Omega(\frac{1}{\epsilon^{1/2-o(1)}})$ . It follows that together, Theorems 5 and Theorem 6 give a lower bound of  $q = \Omega(\frac{1}{\epsilon^{1/2-o(1)}})$  that applies to every choice of  $\epsilon \geq \Omega(\frac{1}{\sqrt{n}})$ . To the best of our knowledge, Theorem 6 is the first lower bound on local list-decoders that is able to prove a lower bound of  $q \geq k$  (and note that this is what we should expect when  $\epsilon < \frac{1}{k}$ ). We also remark that the requirement that  $\epsilon$  is not too small compared to  $n$  (as is made in Theorem 6) is required (as we cannot prove lower bounds on the number of queries in case  $\epsilon < \frac{1}{n}$ ).

Goldreich and Levin [6] showed that locally list-decodable codes with small  $\epsilon < 1/k$  can be used to give constructions of hard-core predicates. We explain this connection in the next section.

## 1.2 Hard-core predicates

The celebrated Goldreich-Levin theorem [6] considers the following scenario: There is a computational task where the required output is *non-Boolean* and is hard to compute on average. We would like to obtain a *hard-core predicate*, which is a *Boolean* value that is hard to compute on average.

The Goldreich-Levin theorem gives a solution to this problem, and in retrospect, the theorem can also be viewed as a  $(\frac{1}{2} - \epsilon, L^{\text{Had}} = O(\frac{1}{\epsilon^2}), q^{\text{Had}} = O(\frac{k}{\epsilon^2}), \delta = \frac{1}{2k})$ -local list-decoder for the Hadamard code, defined by:  $\text{Enc}^{\text{Had}} : \{0, 1\}^k \rightarrow \{0, 1\}^{n=2^k}$ , where for every  $r \in \{0, 1\}^k$ ,

$$\text{Enc}^{\text{Had}}(x)_r = \left( \sum_{i \in [k]} x_i \cdot r_i \right) \bmod 2.$$

In retrospect, the Goldreich-Levin theorem can also be seen as showing that *any* locally list-decodable code with suitable parameters can be used to produce hard-core predicates.

We consider two scenarios for the Goldreich-Levin theorem depending on whether we want to extract a hard-core bit from a function  $g : \{0, 1\}^\ell \rightarrow \{0, 1\}^\ell$  that is *hard to compute* on a random input, or to extract a hard-core bit from a one-way function  $f : \{0, 1\}^\ell \rightarrow \{0, 1\}^\ell$  that is *hard to invert* on a random output.

### 1.2.1 Functions that are hard to compute

Here the goal is to transform a *non-Boolean function*  $g$  that is hard to compute on a random input, into a *predicate*  $g^{\text{pred}}$  that is hard to compute on a random input. More precisely:

- *Assumption:* There is a non-Boolean function that is hard to compute with probability  $\rho$ . Namely, a function  $g : \{0, 1\}^\ell \rightarrow \{0, 1\}^\ell$  such that for every circuit  $C$  of size  $s$ ,

$$\Pr_{x \leftarrow U_\ell} [C(x) = g(x)] \leq \rho.^3$$

- *Conclusion:* There is a predicate  $g^{\text{pred}} : \{0, 1\}^{\ell'} \rightarrow \{0, 1\}$  that is hard to compute with probability  $\frac{1}{2} + \epsilon$ .

Namely, for every circuit  $C'$  of size  $s'$ ,

$$\Pr_{x \leftarrow U_{\ell'}} [C'(x) = g^{\text{pred}}(x)] \leq \frac{1}{2} + \epsilon.$$

<sup>3</sup> We use  $U_\ell$  to denote the uniform distribution on  $\ell$  bits.

- *Requirements:* The goal is to show that for every  $g$ , there exists a function  $g^{\text{pred}}$  with as small an  $\epsilon$  as possible, without significant losses in the other parameters (meaning that  $s'$  is not much smaller than  $s$ , and  $\ell'$  is not much larger than  $\ell$ ).

The Goldreich-Levin theorem for this setting can be expressed as follows.

► **Theorem 7** (Goldreich-Levin for functions that are hard to compute [6]). *For every function  $g : \{0, 1\}^\ell \rightarrow \{0, 1\}^\ell$ , define  $g^{\text{pred}} : \{0, 1\}^{\ell'=2\ell} \rightarrow \{0, 1\}$  by  $g^{\text{pred}}(x, r) = \text{Enc}^{\text{Had}}(g(x))_r$ , and  $\rho = \frac{\epsilon}{2 \cdot L^{\text{Had}}} = \text{poly}(\epsilon)$ . If for every circuit  $C$  of size  $s$ ,*

$$\Pr_{x \leftarrow U_\ell} [C(x) = g(x)] \leq \rho,$$

*then for every circuit  $C'$  of size  $s' = \frac{s}{q^{\text{Had}} \cdot \text{poly}(\ell)} = s \cdot \text{poly}(\frac{\epsilon}{\ell})$ ,*

$$\Pr_{x \leftarrow U_{2\ell}} [C'(x) = g^{\text{pred}}(x)] \leq \frac{1}{2} + \epsilon.$$

The Hadamard code can be replaced by any locally list-decodable code with list size  $L$  for decoding from radius  $\frac{1}{2} - \epsilon$ , with  $q$  queries for  $\delta = 1/(2k)$ . For such a code (assuming also that the local list-decoder can be computed efficiently) one gets the same behavior. Specifically, if the initial function is sufficiently hard and  $\rho = \frac{\epsilon}{2L}$ , then the Boolean target function is hard to compute, up to  $\frac{1}{2} + \epsilon$  for circuits of size roughly  $s' = s/q$ .

#### Is it possible to improve the Goldreich-Levin theorem for $\rho \ll 1/s$ ?

Suppose that we are given a function  $g : \{0, 1\}^\ell \rightarrow \{0, 1\}^\ell$  that is hard to compute for circuits of size  $s = \text{poly}(\ell)$ , with success say  $\rho = 1/2^{\sqrt{\ell}}$ . When applying Theorem 7, we gain nothing compared to the case that  $\rho = 1/\text{poly}(\ell)$ . In both cases, we can obtain  $\epsilon = 1/\text{poly}(\ell)$ , but not smaller! (Since otherwise  $s' = s \cdot \text{poly}(\epsilon/\ell)$  is smaller than 1 and the result is meaningless).

This is disappointing, as we may have expected to obtain  $\epsilon \approx \rho = 1/2^{\sqrt{\ell}}$ , or at least, to gain over the much weaker assumption that  $\rho = 1/\text{poly}(\ell)$ . This leads to the following open problem:

► **Open problem 8** (Improve Goldreich-Levin for functions that are hard to compute). *Suppose we are given a function  $g : \{0, 1\}^\ell \rightarrow \{0, 1\}^\ell$  such that circuits  $C$  of size  $s = \text{poly}(\ell)$  cannot compute  $g$  with success  $\rho = 1/2^{\sqrt{\ell}}$ . Is it possible to convert  $g$  into a predicate with hardness  $\frac{1}{2} + \epsilon$  for  $\epsilon = 1/\ell^{\omega(1)}$ ?*

This is not possible to achieve using the Hadamard code, because the number of queries is  $q \geq 1/\epsilon$ , and Theorem 7 requires  $s \geq s' \cdot q \geq q \geq 1/\epsilon$ , which dictates that  $\epsilon \geq 1/s$ .

Note that when  $\rho$  is small, we can afford list-decodable codes with huge list sizes of  $L \approx 1/\rho$ . Motivated by this observation, we can ask the following question:

Is it possible to solve this open problem by substituting the Hadamard code with a better code? Specifically, is it possible for local list-decoders to have  $q = \frac{1}{\epsilon^{o(1)}}$  if allowed to use *huge* lists of size say  $2^{\sqrt{k}}$ , approaching the trivial bound  $2^k$ ? (Note that in the Hadamard code, the list size used is  $\text{poly}(1/\epsilon) = \text{poly}(k)$  which is exponentially smaller).

We show, in Theorem 6, that it is impossible to solve the open problem by replacing the Hadamard code with a different locally list-decodable code.

The natural next question is whether we can use other techniques (not necessarily local list-decoding) to achieve the goal stated above. In this paper, we show that this cannot be done by *black-box techniques*:

► **Informal Theorem 9** (Black-box impossibility result for functions that are hard to compute). *If  $\rho \geq \frac{1}{2^{\ell/3}}$ ,  $s = 2^{o(\ell)}$  is larger than some fixed polynomial in  $\ell$ , and  $\epsilon = \frac{1}{s^{\omega(1)}}$ , then there does not exist a map that converts a function  $g$  into a function  $g^{\text{pred}}$  together with a black-box reduction showing that  $g^{\text{pred}}$  is a hard-core predicate for  $g$ .*

The parameters achieved in Theorem 9 rule out black-box proofs in which  $\epsilon = \frac{1}{s^{\omega(1)}}$ , not only for  $s = \text{poly}(\ell)$  and  $\rho = 2^{-\sqrt{\ell}}$  (as in Open problem 8) but also for  $\rho = 2^{-\Omega(\ell)}$ , and allowing much larger  $s$  as long as  $s = 2^{o(\ell)}$ .

The precise statement of Theorem 9 is stated in Theorem 19, and the precise model is explained in Section 3.1.

To the best of our knowledge, this is the first result of this kind, that shows black-box impossibility results for Open problem 8. Moreover, we believe that the model that we introduce in Section 3.1 is very general and captures all known black-box techniques. In particular, our model (which we view as a conceptual contribution) allows the reduction to introduce nonuniformity when converting an adversary  $C'$  that breaks  $g^{\text{pred}}$  into an adversary  $C$  that breaks  $g$ . See discussion in Remark 14.

### 1.2.2 Functions that are hard to invert

Here the goal is to transform a one-way function  $f$  into a new one-way function  $f^{\text{newOWF}}$  and a predicate  $f^{\text{pred}}$  such that it is hard to compute  $f^{\text{pred}}(x)$  given  $f^{\text{newOWF}}(x)$ . More precisely:

- *Assumption:* There is a one-way function that is hard to invert with probability  $\rho$ .

Namely, a function  $f : \{0, 1\}^\ell \rightarrow \{0, 1\}^\ell$  such that for every circuit  $C$  of size  $s$ ,

$$\Pr_{x \leftarrow U_\ell} [C(f(x)) \in f^{-1}(f(x))] \leq \rho.$$

- *Conclusion:* There is a one-way function  $f^{\text{newOWF}} : \{0, 1\}^{\ell'} \rightarrow \{0, 1\}^{\ell'}$ , and a predicate  $f^{\text{pred}} : \{0, 1\}^{\ell'} \rightarrow \{0, 1\}$ , such that it is hard to predict  $f^{\text{pred}}(x)$  with advantage  $\frac{1}{2} + \epsilon$ , when given access to  $f^{\text{newOWF}}(x)$ .

Namely, for every circuit  $C'$  of size  $s'$ ,

$$\Pr_{x \leftarrow U_{\ell'}} [C'(f^{\text{newOWF}}(x)) = f^{\text{pred}}(x)] \leq \frac{1}{2} + \epsilon.$$

- The goal is to show that for every  $f$ , there exist functions  $f^{\text{newOWF}}, f^{\text{pred}}$  with as small  $\epsilon$  as possible, without significant losses in the other parameters (meaning that:  $s'$  is not much smaller than  $s$ , and  $\ell'$  is not much larger than  $\ell$ ).

The Goldreich-Levin theorem for this setting can be expressed as follows.

► **Theorem 10** (Goldreich-Levin for functions that are hard to invert). *For a function  $f : \{0, 1\}^\ell \rightarrow \{0, 1\}^\ell$ , define  $f^{\text{newOWF}} : \{0, 1\}^{2\ell} \rightarrow \{0, 1\}^{2\ell}$  by  $f^{\text{newOWF}}(x, r) = (f(x), r)$ ,  $f^{\text{pred}} : \{0, 1\}^{2\ell} \rightarrow \{0, 1\}$  by  $f^{\text{pred}}(x, r) = \text{Enc}^{\text{Had}}(x)_r$ , and  $\rho = \frac{\epsilon}{2 \cdot L^{\text{Had}}} = \text{poly}(\epsilon)$ . If for every circuit  $C$  of size  $s$ ,*

$$\Pr_{x \leftarrow U_\ell} [C(f(x)) \in f^{-1}(f(x))] \leq \rho,$$

*then for every circuit  $C'$  of size  $s' = \frac{s}{q^{\text{Had}} \cdot \text{poly}(\ell)} = s \cdot \text{poly}(\frac{\epsilon}{\ell})$ ,*

$$\Pr_{x \leftarrow U_{2\ell}} [C'((f^{\text{newOWF}}(x))) = f^{\text{pred}}(x)] \leq \frac{1}{2} + \epsilon.$$



► **Remark 11.** The problem of obtaining a hard-core predicate for one-way functions, is interesting only if an unbounded adversary  $\phi : \{0, 1\}^{\ell'} \rightarrow \{0, 1\}$  can predict  $f^{\text{pred}}(x)$  when given  $f^{\text{newOWF}}(x)$  as input. If this is not required, then one can take  $\ell' = \ell + 1$ ,  $f^{\text{pred}}(x) = x_1$ , and  $f^{\text{newOWF}}(x_1, \dots, x_{n+1}) = f(x_2, \dots, x_{n+1})$ . However, this is trivial, and is not useful in applications. Therefore, when considering this problem, we will assume that there exists such a  $\phi : \{0, 1\}^{\ell'} \rightarrow \{0, 1\}$ .

A natural example is the case where the original one-way function  $f$  and the constructed function  $f^{\text{newOWF}}$  are one-way permutations. In fact, in the case that  $f, f^{\text{newOWF}}(x)$  are permutations, the setup of “functions that are hard to invert” can be seen as a special case of the setup of functions that are “hard to compute” by taking  $g = f^{-1}$ , and  $g^{\text{pred}}(y) = f^{\text{pred}}((f^{\text{newOWF}})^{-1}(y))$ .

We point out that, in this setting, the circuit  $C'$  that is trying to invert  $f$  (that is, to compute  $g$ ) has an advantage over its counterpart in the setup of “functions that are hard to compute”: It can use the efficient algorithm that computes the “forward direction” of  $f$ , when trying to invert  $f$ . In terms of  $g$ , this means that the circuit  $C'$  can compute  $g^{-1}$  for free.

### Is it possible to improve the Goldreich-Levin theorem for $\rho \ll 1/s$ ?

The same problem that we saw with functions that are hard to compute, also shows up in the setup of functions that are hard to invert. Suppose that we are given a function  $f : \{0, 1\}^{\ell} \rightarrow \{0, 1\}^{\ell}$  that is hard to invert for circuits of size  $s = \text{poly}(\ell)$  with success, say,  $\rho = 1/2^{\sqrt{\ell}}$ . When applying Theorem 10, we gain nothing compared to the case that  $\rho = 1/\text{poly}(\ell)$ . In both cases, we can obtain  $\epsilon = 1/\text{poly}(\ell)$ , but not smaller! This is expressed in the next open problem:

► **Open problem 12** (Improve Goldreich-Levin for functions that are hard to invert). *If we are given a one-way function  $f : \{0, 1\}^{\ell} \rightarrow \{0, 1\}^{\ell}$  such that circuits  $C$  of size  $s = \text{poly}(\ell)$  cannot invert  $f$  with success  $\rho = 1/2^{\sqrt{\ell}}$ . Is it possible to obtain a hard-core predicate  $f^{\text{pred}}$  with hardness  $\frac{1}{2} + \epsilon$  for  $\epsilon = 1/\ell^{\omega(1)}$  for some choice of one-way function  $f^{\text{newOWF}}$ ?*

In this paper, we show that this cannot be done by *black-box techniques*. The formulation of Theorem 13 below, is very similar to that of Theorem 9 with some small modification in the parameters.

► **Informal Theorem 13** (Black-box impossibility result for functions that are hard to invert). *If  $\rho \geq 2^{-\ell/5}$ ,  $s = 2^{o(\ell)}$  is larger than some fixed polynomial in  $\ell$ , and  $\epsilon = \frac{1}{s^{\omega(1)}}$  then there does not exist a map that converts a function  $f$  into functions  $f^{\text{newOWF}}, f^{\text{pred}}$  together with a black-box reduction showing that  $f^{\text{pred}}$  is a hard-core predicate for  $f^{\text{newOWF}}$ .*

To the best of our knowledge, this is the first result of this kind, that shows black-box impossibility results for open problem 12. Moreover, we believe that the model that we introduce is very general, and captures all known black-box techniques. In particular, our model (which we view as a conceptual contribution) allows the reduction to compute the easy direction of the function  $f$ , and to introduce nonuniformity when converting an adversary  $C'$  that breaks  $f^{\text{pred}}$  into an adversary  $C$  that breaks  $f$ .

► **Remark 14** (The model we use for black-box proofs). Many different models of “black-box techniques” for cryptographic primitives were studied in the literature and the reader is referred to [12] for a discussion and a taxonomy. The model for “black-box technique” that we use is described in detail in Section 3. The notion that we use is incomparable to the ones discussed in [12], specifically:

- We require that there is a “transformation map” which given any function  $f$  produces functions  $f^{\text{newOWF}}$  and  $f^{\text{pred}}$ , however, unlike [12], we do not make the requirement that this transformation map can be efficiently computed.
- We require that there is a reduction  $\text{Red}$  such that for any  $f$ , and for every adversary  $C'$  (not necessarily efficient) breaking the security of  $f^{\text{pred}}$ ,  $\text{Red}^{f,C'}$  can be used to invert  $f$ .
- However, we give reductions more power: We also allow  $\text{Red}$  to introduce nonuniformity (that could depend on  $C'$  and  $f$ ). Formally, for every adversary  $C'$  that breaks the security of  $f^{\text{pred}}$ , we require that there exists a short nonuniform advice string  $\alpha$  such that  $\text{Red}^{C',f}(\cdot, \alpha)$  inverts  $f$ .

### 1.3 More related work

#### Lower bounds on the number of queries of local decoders for *uniquely* decodable codes

In this paper, we prove lower bounds on the number of queries of local list-decoders. There is a long line of work that is concerned with proving lower bounds on the number of queries of uniquely decodable codes. As we have explained in Section 1.1.1, the parameter regime considered in the setting of uniquely decodable codes is very different than the parameter regime we consider here [18].

#### Lower bounds on nonuniform black-box reductions for hardness amplification

A problem that is closely related to proving lower bounds on the number of queries of local list-decoders is the problem of proving lower bounds on the number of queries of nonuniform black-box reductions for hardness amplification. We have already discussed this line of work [17, 15, 3, 7, 14] in Section 1.1.1.

Lower bounds on such reductions can be translated to lower bounds on local list-decoders (as long as the number of coins tossed by the local list-decoders is small). We remark that for the purpose of hardness amplification, it does not make sense to take codes with small rate (namely, codes with  $n = 2^{k^{\Omega(1)}}$ ). The focus of Theorem 5 is to handle such codes.

Additionally, when using codes for hardness amplification, it does not make sense to take  $\epsilon < 1/k$  (or even  $\epsilon < 1/\sqrt{k}$ ). In contrast, the parameter regime considered in Theorem 6 focuses on small  $\epsilon$ .

Motivated by hardness amplification, there is also a related line of work studying limitations on the complexity of local list decoders (and specifically, whether these decoders need to compute the majority function) [17, 15, 9, 3, 7, 14].

Another approach to prove limitations on hardness amplification is to show that assuming certain cryptographic assumptions, hardness amplification that is significantly better than what is currently known is impossible, see e.g., [5] for a discussion.

#### Other improvements of the Goldreich-Levin theorem

In this paper, we are interested in whether the Goldreich-Levin theorem can be improved. Specifically, we are interested in improvements where, when the original function has hardness  $\rho = 2^{-\Omega(\ell)}$  for polynomial size circuits, then the hard-core predicate has hardness  $\frac{1}{2} + \epsilon$  for  $\epsilon = \ell^{-\omega(1)}$ . We remark that there are other aspects of the Goldreich-Levin theorem that one may want to improve.

- When given an initial non-Boolean function on  $\ell$  bits, the Goldreich-Levin theorem produces a hard-core predicate on  $\ell' = 2\ell$  bits. It is possible to make  $\ell'$  smaller (specifically,  $\ell' = \ell + O(\log(1/\epsilon))$ ) by using other locally list-decodable codes instead of Hadamard. Our limitations apply to *any* construction (even one that is not based on codes) and in particular also for such improvements.
- It is sometimes desirable to produce many hard-core bits (instead of the single hard-core bit) that is obtained by a hard-core predicate. This can be achieved by using “extractor codes” with a suitable local list-decoding algorithm. The reader is referred to [16] for more details. Once again, our limitations obviously apply also for the case of producing many hard-core bits.

## Organization of the paper

We give a high level overview of our technique in Section 2. Some of our results on hard-core predicates appear in Section 3 (which includes a precise description of the model and formal restatements of Theorem 9). We refer the interested reader to the full version [13] for a precise description of the model and formal restatement of Theorem 13. Section 4 contains some concluding remarks and open problems. All proofs can be found in the full version.

## 2 Technique

In this section we give a high level overview of our technique. Our approach builds on earlier work for proving lower bounds on the number of queries or reductions for hardness amplification [17, 15, 7]. In this section, we give a high level overview of the arguments used to prove our main theorems.

### 2.1 Local list-decoders on random noisy codewords

Following [17, 15, 7], we will consider a scenario which we refer to as “random noisy codewords” in which a uniformly chosen message  $m$  is encoded, and the encoding is corrupted by a binary symmetric channel.

► **Definition 15** (Binary symmetric channels). *Let  $\text{BSC}_p^n$  be the experiment in which a string  $Z \in \{0, 1\}^n$  is sampled, where  $Z = Z_1, \dots, Z_n$  is composed of i.i.d. bits, such that for every  $i \in [n]$ ,  $\Pr[Z_i = 1] = p$ .*

► **Definition 16** (Random noisy codewords). *Given a function  $\text{Enc} : \{0, 1\}^k \rightarrow \{0, 1\}^n$  and  $p > 0$  we consider the following experiment (which we denote by  $\text{RNSY}_p^{\text{Enc}}$ ):*

- *A message  $m \leftarrow \{0, 1\}^k$  is chosen uniformly.*
- *A noise string  $z \leftarrow \text{BSC}_p^n$  is chosen from a binary symmetric channel.*
- *We define  $w = \text{Enc}(m) \oplus z$ .*

*We use  $(m, z, w) \leftarrow \text{RNSY}_p^{\text{Enc}}$  to denote  $m, z, w$  which are sampled by this experiment. We omit  $\text{Enc}$  if it is clear from the context.*

Our goal is to prove lower bounds on the number of queries  $q$  of a  $(\frac{1}{2} - \epsilon, L, q, \delta)$ -local list-decoder  $\text{Dec}$  for a code  $\text{Enc} : \{0, 1\}^k \rightarrow \{0, 1\}^n$ . For this purpose, we will consider the experiment  $\text{RNSY}_p$  for the values  $p = \frac{1}{2} - 2\epsilon$  and  $p = \frac{1}{2}$ .

For  $p = \frac{1}{2} - 2\epsilon$ , and  $(m, z, w) \leftarrow \text{RNSY}_{\frac{1}{2}-\epsilon}$ , by a Chernoff bound, the Hamming weight of  $z$  is, with very high probability, less than  $\frac{1}{2} - \epsilon$ . This implies that  $\text{dist}(w, \text{Enc}(m)) \leq \frac{1}{2} - \epsilon$ , meaning that  $m \in \text{List}_{\frac{1}{2}-\epsilon}^{\text{Enc}}(w)$ . It follows that there must exist  $j \in [L]$  such that when given input  $j$ , and oracle access to  $w$ , the decoder  $\text{Dec}$  recovers the message  $m$ .

For  $p = \frac{1}{2}$ , and  $(m, z, w) \leftarrow \text{RNSY}_{\frac{1}{2}}$ , the string  $z$  is uniformly distributed and independent of  $m$ . This means that  $w = \text{Enc}(m) \oplus z$  is uniformly distributed and independent of  $m$ . Consequently, when Dec is given oracle access to  $w$ , there is no information in  $w$  about the message  $m$ , and so, for every  $j \in [L]$ , the probability that Dec recovers  $m$  when given input  $j$  and oracle access to  $w$  is exponentially small.

Loosely speaking, this means that Dec can be used to distinguish  $\text{BSC}_{\frac{1}{2}-2\epsilon}^n$  from  $\text{BSC}_{\frac{1}{2}}^n$ . It is known that distinguishing these two distributions requires many queries. We state this informally below.

► **Informal Theorem 17.** *Any function  $T : \{0, 1\}^q \rightarrow \{0, 1\}$  that distinguishes  $\text{BSC}_{\frac{1}{2}-2\epsilon}^q$  from  $\text{BSC}_{\frac{1}{2}}^q$  with advantage  $\delta$ , must have  $q = \Omega(\frac{\log(1/\delta)}{\epsilon^2})$ .*

Thus, in order to prove a tight lower bound of  $q = \Omega(\frac{\log(1/\delta)}{\epsilon^2})$ , it is sufficient to show how to convert a  $(\frac{1}{2} - \epsilon, L, q, \delta)$ -local list-decoder Dec, into a function  $T$  that distinguishes  $\text{BSC}_{\frac{1}{2}-2\epsilon}^q$  from  $\text{BSC}_{\frac{1}{2}}^q$  with advantage  $\delta$ . Note that we can allow  $T$  to be a “randomized procedure” that tosses coins, as by an averaging argument, such a randomized procedure can be turned into a deterministic procedure.

## 2.2 Warmup: the case of unique decoding

Let us consider the case that  $L = 1$  (that is unique decoding). We stress that this case is uninteresting, as by the Plotkin bound, it is impossible for nontrivial codes to be uniquely decodable for  $\epsilon < \frac{1}{4}$ , and so, there are no local decoders for  $L = 1$  and  $\epsilon < \frac{1}{4}$ , regardless of the number of queries. Nevertheless, this case will serve as a warmup for the approach we use later.

Our goal is to convert Dec into a randomized procedure  $T : \{0, 1\}^q \rightarrow \{0, 1\}$  that distinguishes  $\text{BSC}_{\frac{1}{2}-2\epsilon}^q$  from  $\text{BSC}_{\frac{1}{2}}^q$ . The procedure  $T$  will work as follows: On input  $x \leftarrow \{0, 1\}^q$ , we choose  $m \leftarrow \{0, 1\}^k$ , and  $i \leftarrow [k]$ . We then run Dec on input  $i$ , and when Dec makes its  $t$ 'th query  $\ell_t \in [n]$  to the oracle, we answer it by  $\text{Enc}(m)_{\ell_t} \oplus x_t$ . That is, we answer as if Dec is run with input  $i$  and oracle access to  $w = \text{Enc}(m) \oplus z$ , for  $z$  chosen from a binary symmetric channel. The final output of  $T$  is whether Dec reproduced  $m_i$ . This procedure  $T$  simulates  $\text{Dec}^w(i)$ , and therefore distinguishes  $\text{BSC}_{\frac{1}{2}-2\epsilon}^q$  from  $\text{BSC}_{\frac{1}{2}}^q$ , implying the desired lower bound.

Both Theorem 5 and Theorem 6 will follow by modifying the basic approach to handle  $L > 1$ . In the remainder of this section, we give a high level overview of the methods that we use. The formal section of this paper does not build on this high level overview, and readers can skip this high level overview and go directly to the formal section if they wish.

## 2.3 Reducing to the coin problem for $\text{AC}^0$

We start with explaining the approach of proving Theorem 6. Consider a randomized procedure  $C$  that on input  $z \in \{0, 1\}^n$ , chooses  $m \leftarrow \{0, 1\}^k$  and prepares  $w = \text{Enc}(m) \oplus z$ . The procedure then computes  $\text{Dec}^w(i, j)$  for all choices of  $i \in [k]$  and  $j \in [L]$  and accepts if there exists a  $j \in [L]$  such that  $\text{Dec}^w(\cdot, j)$  recovers  $m$ . By the same rationale as in Section 2.2,  $C$  distinguishes  $\text{BSC}_{\frac{1}{2}-2\epsilon}^n$  from  $\text{BSC}_{\frac{1}{2}}^n$ . This does not seem helpful, because  $C$  receives  $n$  input bits, and we cannot use Theorem 17 to get a lower bound on  $q$ .

Inspired by a lower bound on the size of nondeterministic reductions for hardness amplification due to Applebaum et al. [2], we make the following observation: The procedure  $C$  can be seen as  $k \cdot L$  computations (one for each choice of  $i \in [k]$  and  $j \in [L]$ ) such that:

- These  $k \cdot L$  computations can be run *in parallel*.
- Once these computations are made, the final answer  $C(z)$  is computed by a constant-depth circuit.
- Each of the  $k \cdot L$  computations makes  $q$  queries into  $z$ , and therefore can be simulated by a size  $O(q \cdot 2^q)$  circuit of depth 2.

Overall, this means that we can implement  $C$  by a circuit of size  $s = \text{poly}(k, L, 2^q)$  and constant depth. (In fact, a careful implementation gives depth 3).

This is useful because there are lower bounds showing that small constant-depth circuits cannot solve the “coin problem”. Specifically, by the results of Cohen, Ganor and Raz [4] circuits of size  $s$  and depth  $d$  cannot distinguish  $\text{BSC}_{\frac{1}{2}-2\epsilon}^n$  from  $\text{BSC}_{\frac{1}{2}}^n$  with constant advantage, unless  $s \geq \exp(\Omega(\frac{1}{\epsilon^{d-1}}))$ .<sup>4</sup> This gives the bound stated in Theorem 6.

We find it surprising that an *information theoretic* lower bound on the number of queries of local list-decoders is proven by considering concepts like constant-depth circuits from *circuit complexity*.

### Extending the argument to lower bounds on hard-core predicates

It turns out that this argument is quite versatile, and this is the approach that we use to prove Theorems 9 and 13. Loosely speaking, in these theorems, we want to prove a lower bound on the number of queries made by a reduction that, when receiving oracle access to an adversary that breaks the hard-core predicate, is able to compute (or invert) the original function too well. Such lower bounds imply that such reductions do not produce small circuits when used in black-box proofs for hard-core predicates.

We will prove such lower bounds by showing that a reduction that makes  $q$  queries can be used to construct a circuit of size  $s \approx 2^q$  and constant depth that solves the coin problem. Interestingly, this argument crucially relies on the fact that constant-depth circuits *can* distinguish  $\text{BSC}_{\epsilon}^n$  from  $\text{BSC}_{2\epsilon}^n$  with size  $\text{poly}(n/\epsilon)$  which follows from the classical results of Ajtai on constant depth circuits for approximate majority [1].<sup>5</sup>

## 2.4 Conditioning on a good $j$

A disadvantage of the approach based on the coin problem is that at best, it can give lower bounds of  $q = \Omega(1/\sqrt{\epsilon})$ , and cannot give tight lower bounds of the form  $q = \Omega(\frac{\log(1/\delta)}{\epsilon^2})$ . In order to achieve such a bound (as is the case in Theorem 5) we will try to reduce to Theorem 17 which *does* give a tight bound in case  $\epsilon$  is not too small.

Our approach builds on the earlier work of Grinberg, Shaltiel and Viola [7] that we surveyed in Section 1.1.1. When given a  $(\frac{1}{2} - \epsilon, L, q, \delta)$ -local list-decoder  $\text{Dec}$ , we say that an index  $j \in [L]$  is *decoding*, if in the experiment  $(m, z, w) \in \text{RNSY}_{\frac{1}{2}-2\epsilon}$ , when  $\text{Dec}$  is given oracle access to  $w$  and input  $j$ , then with probability  $1 - 10\delta$  over  $i \in [k]$ , we have that  $\text{Dec}^w(i, j)$  recovers  $m_i$ .

<sup>4</sup> These results of [4] improve upon earlier work of Shaltiel and Viola [15] that gave slightly worse bound. These results are tight as shown by Limaye et al. [10] (that also extended the lower bound to hold for circuits that are also allowed to use parity gates).

<sup>5</sup> The proof of Theorem 13 uses an additional versatility of the argument (which we express in the terminology of codes): The argument works even if the individual procedures that are run in parallel are allowed to have some limited access to the message  $m$ , as long as this does not enable them to recover  $m$ . This property is used to handle reductions in a cryptographic setup, where reductions have access to the easy direction of a one-way function.

We use a careful averaging argument to show that there exists an index  $j' \in [L]$ , and a fixed choice of the random coins of Dec, such that  $j'$  is decoding with probability at least  $\Omega(1/L)$ . We then consider the experiment  $\text{RNSY}'_{\frac{1}{2}-2\epsilon}$  in which we choose  $(m, z, w) \leftarrow \text{RNSY}'_{\frac{1}{2}-2\epsilon}$  *conditioned* on the event  $\{j' \text{ is decoding}\}$ .

We have made progress, because in the experiment  $\text{RNSY}'_{\frac{1}{2}-2\epsilon}$  there exists a unique  $j'$  that is decoding, and so, when we implement the strategy explained in Section 2.2 we only need to consider this *single*  $j'$ , which intuitively means that our scenario is similar to the warmup scenario of unique decoding described in Section 2.2.

The catch is that when choosing  $(m, z, w) \leftarrow \text{RNSY}'_{\frac{1}{2}-2\epsilon}$ , we no longer have that  $z$  is distributed like  $\text{BSC}_{\frac{1}{2}-2\epsilon}^n$  (as the distribution of  $z$  may be skewed by conditioning on the event that  $j'$  is decoding).

Shaltiel and Viola [15] (and later work [7, 14]) developed tools to handle this scenario. Loosely speaking, using these tools, it is possible to show that a large number of messages  $m$  are “useful” in the sense that there exists an event  $A_m$  such that if we consider  $(m, z, w)$  that are chosen from  $\text{RNSY}'_{\frac{1}{2}-2\epsilon}$  *conditioned* on  $A_m$ , then there exists a subset  $B(m) \subseteq [n]$  of small size  $b$ , such that  $z_{B(m)}$  is fixed, and  $z_{[n] \setminus B(m)}$  is distributed like  $\text{BSC}_{\frac{1}{2}-2\epsilon}^{n-b}$ .

If the number of possible choices for sets  $B(m)$  is small, then by the pigeon-hole principle, there exists a fixed choice  $B$  that is good for a large number of useful messages  $m$ . This can be used to imitate the argument we used in the warmup, and prove a lower bound.<sup>6</sup>

### Extending the argument to the case of small rate

A difficulty, that prevented [7] from allowing length as large as  $n = 2^k$ , is that  $B(m)$  is a subset of  $[n]$ , and so, even if  $b = |B| = 1$ , the number of possible choices for such sets is at least  $n$ . For the pigeon-hole principle argument above, we need that the number of messages (that is  $2^k$ ) is much larger than the number of possible choices for  $B(m)$  (which is at least  $n$ ). This means that one can only handle  $n$  which is sufficiently smaller than  $2^k$ , and this approach cannot apply to codes with small rate (such as the Hadamard code).

We show how to solve this problem, and prove lower bounds for small rate codes. From a high level, our approach can be explained as follows: We consider the distribution of  $B(m) = \{Y_1(m) < \dots < Y_b(m)\}$  for a uniformly chosen useful  $m$ . We first show that if all the  $Y_j$ 's have large min-entropy, then it is possible to prove a lower bound on  $q$  by reducing to Theorem 17 (the details of this are explained in the actual proof).

If on the other hand, one of the  $Y_j$ 's has low min-entropy, then we will restrict our attention to a subset of useful messages on which  $Y_j$  is fixed. Loosely speaking, this reduces  $b$  by one, while not reducing the number of useful messages by too much (because the low min-entropy condition says that the amount of information that  $Y_j$  carries on  $m$  is small). In this trench warfare, in every iteration, we lose a fraction of useful messages, for the sake of decreasing  $b$  by one. Thus, eventually, we either reach the situation that all the  $Y_j$ 's have large min-entropy, in which case we are done, or we reach the situation where  $B$  is fixed for all messages which we can also handle by the above.

We can withstand the losses and eventually win if  $\epsilon$  is sufficiently larger than  $1/\sqrt{k}$ .

<sup>6</sup> Loosely speaking, this is because for good messages, in the conditioned experiment,  $z$  is distributed like  $\text{BSC}_{\frac{1}{2}-2\epsilon}^n$  (except that some bits of  $z$  are fixed as a function of  $m$ ). Furthermore, as there are many good messages, the local list-decoder does not have enough information to correctly recover the message when given oracle access to  $\text{Enc}(m) \oplus \text{BSC}_{\frac{1}{2}}^n = \text{BSC}_{\frac{1}{2}}^n$ .

### 3 Limitations on black-box proofs for hard-core predicates

In this section, we present some of our results regarding the limitations on black-box proofs for hard-core predicate theorems. Specifically, we state our results for functions that are hard to compute, give a formal restatement of Theorem 9. Due to space limitation, the formal model and precise statement of results for the case of functions that are hard to invert appear in the full version [13].

#### 3.1 The case of functions that are hard to compute

##### 3.1.1 The model for black-box proofs

In this section, we state and explain our model for black-box proofs for hard core predicates, in the setting of functions that are hard to compute. The formal definition is given in Definition 18. Below, we provide a detailed explanation for the considerations made while coming up with the formal definition. The reader can skip directly to the formal definition if he wishes.

##### Explanation of the model

Recall that (as explained in Section 1.2.1) the Goldreich-Levin theorem (stated precisely in Theorem 7) has the following form:

- We are given an arbitrary hard function  $g : \{0, 1\}^\ell \rightarrow \{0, 1\}^\ell$ . (Intuitively, it is assumed that it is hard to compute  $g$  with success probability  $\rho$ ).
- There is a specified construction that transforms  $g$  into a predicate  $g^{\text{pred}} : \{0, 1\}^{\ell'} \rightarrow \{0, 1\}$  for some  $\ell'$  related to  $\ell$ . (Intuitively, we will want to argue that  $g^{\text{pred}}$  is a hard-core predicate that is hard to compute with success  $\frac{1}{2} + \epsilon$ ).

We will model this construction as a map  $\text{Con}$ , which, given  $g$  produces  $g^{\text{pred}}$ . We place *no limitations* on the map  $\text{Con}$  (and, in particular, do not require that  $g^{\text{pred}}$  can be efficiently computed if  $g$  is). This only makes our results stronger.

In the case of Theorem 7, we have that:  $\text{Con}(g) = g^{\text{pred}}$  where  $\ell' = 2\ell$  and we think of the  $\ell'$ -bit long input of  $g^{\text{pred}}$  as two strings  $x, r \in \{0, 1\}^\ell$ , setting:

$$g^{\text{pred}}(x, r) = \text{Enc}^{\text{Had}}(g(x))_r = \left( \sum_{i \in [\ell]} g(x)_i \cdot r_i \right) \bmod 2.$$

- We model the proof showing that  $g^{\text{pred}}$  is a hard-core predicate in the following way: The proof is a pair  $(\text{Con}, \text{Red})$  where  $\text{Red}^{(\cdot)}$  is an oracle procedure, such that when  $\text{Red}^{(\cdot)}$  receives oracle access to an “adversary”  $h : \{0, 1\}^{\ell'} \rightarrow \{0, 1\}$  that breaks the security of  $g^{\text{pred}}$ , we have that  $\text{Red}^h$  breaks the security of  $g$ . More precisely, we require that: for every  $g : \{0, 1\}^\ell \rightarrow \{0, 1\}^\ell$  and for every  $h : \{0, 1\}^{\ell'} \rightarrow \{0, 1\}$  such that:

$$\Pr_{x \leftarrow U_{\ell'}} [h(x) = g^{\text{pred}}(x)] \geq \frac{1}{2} + \epsilon,$$

it holds that:

$$\Pr_{x \leftarrow U_\ell} [\text{Red}^h(x) = g(x)] \geq \rho.$$

- In the actual definition, we will allow the reduction to have more power (which only makes our results stronger). As we are aiming to prove a result on circuits (which are allowed to use nonuniform advice) we will allow the reduction to receive an advice



string  $\alpha$  of length  $t$ , where, this advice string can depend on  $g$  and  $h$ . This leads to the following strengthening of the requirement above. Namely, we will require that: for every  $g : \{0, 1\}^\ell \rightarrow \{0, 1\}^\ell$  and for every  $h : \{0, 1\}^{\ell'} \rightarrow \{0, 1\}$ , that:

$$\Pr_{x \leftarrow U_{\ell'}} [h(x) = g^{\text{pred}}(x)] \geq \frac{1}{2} + \epsilon,$$

there exists  $\alpha \in \{0, 1\}^t$  such that:

$$\Pr_{x \leftarrow U_\ell} [\text{Red}^h(x, \alpha) = g(x)] \geq \rho.$$

We remark that in many related settings (for example, “hardness amplification”; see [15, 7], for a discussion) known proofs by reduction *critically make use* of the ability to introduce nonuniformity, and so, we feel that when ruling out black-box proofs in scenarios involving circuits, it is necessary to consider nonuniform black-box reductions.

- We make no restrictions on the complexity of the procedure  $\text{Red}^{(\cdot)}$ , except for requiring that it makes at most  $q$  queries to its oracle (for some parameter  $q$ ). Our black-box impossibility results will follow from proving lower bounds on  $q$ .

### Formal definition

We now give a formal definition of our model for black-box proofs for hard-core predicates.

► **Definition 18** (Nonuniform black-box proofs for hard-core predicates for hard-to-compute functions). *A pair  $(\text{Con}, \text{Red})$  is a nonuniform black-box proof for hard-core predicates for hard-to-compute functions with parameters  $\ell, \ell', \rho, \epsilon$ , that uses  $q$  queries, and  $t$  bits of advice if:*

- *Con is a construction map which given a function  $g : \{0, 1\}^\ell \rightarrow \{0, 1\}^\ell$ , produces a function  $\text{Con}(g) = g^{\text{pred}}$ , where  $g^{\text{pred}} : \{0, 1\}^{\ell'} \rightarrow \{0, 1\}$ .*
- *$\text{Red}^{(\cdot)}$  is a reduction, that is an oracle procedure that, given oracle access to a function  $h : \{0, 1\}^{\ell'} \rightarrow \{0, 1\}$ , makes at most  $q$  queries to its oracle.*

*Furthermore, for every functions  $g : \{0, 1\}^\ell \rightarrow \{0, 1\}^\ell$  and  $h : \{0, 1\}^{\ell'} \rightarrow \{0, 1\}$  such that:*

$$\Pr_{x \leftarrow U_{\ell'}} [h(x) = g^{\text{pred}}(x)] \geq \frac{1}{2} + \epsilon,$$

*there exists  $\alpha \in \{0, 1\}^t$ , such that:*

$$\Pr_{x \leftarrow U_\ell} [\text{Red}^h(x, \alpha) = g(x)] \geq \rho.$$

### The role of the number of queries, and black-box impossibility results

We now explain the role of the parameter  $q$  (that measures the number of queries made by  $\text{Red}$ ) and why lower bounds on  $q$  translate into black-box impossibility results.

For this purpose, it is illustrative to examine the argument showing that nonuniform black-box proofs yield hard-core predicates: When given a pair  $(\text{Con}, \text{Red})$  that is a nonuniform black-box proof for hard-core predicates for hard-to-compute functions with parameters  $\ell, \ell', \rho, \epsilon$ , that uses  $q$  queries, and  $t$  bits of advice, we obtain that for any function  $g : \{0, 1\}^\ell \rightarrow \{0, 1\}^\ell$ , if there exists a circuit  $C' : \{0, 1\}^{\ell'} \rightarrow \{0, 1\}$  of size  $s'$  such that:

$$\Pr_{x \leftarrow U_{\ell'}} [C'(x) = g^{\text{pred}}(x)] \geq \frac{1}{2} + \epsilon,$$

then there exists  $\alpha \in \{0, 1\}^t$ , such that:

$$\Pr_{x \leftarrow U_\ell} [\text{Red}^{C'}(x, \alpha) = g(x)] \geq \rho.$$

Note that if the reduction  $\text{Red}$  can be implemented by a circuit of size  $r$ , then the circuit  $C(x) = \text{Red}^{C'}(x, \alpha)$  is a circuit of size:

$$s = r + t + q \cdot s'$$

that computes  $g$  with success probability  $\rho$ .

It follows that in a black-box proof, with  $q$  queries, and  $t$  bits of advice, we get a hard-core theorem that needs to assume that the original function  $g$  has hardness against circuits of size  $s$ , for:

$$s \geq q + t.$$

### 3.1.2 Precise statements of limitations

Our main result on black-box proofs for hard-core predicates in the setting of functions that are hard to compute is the following theorem.

► **Theorem 19.** *There exists a universal constant  $\beta > 0$  such that for every sufficiently large  $\ell$  and  $\ell'$  we have that if  $(\text{Con}, \text{Red})$  is a nonuniform black-box proof for hard-core predicates for hard-to-compute functions with parameters  $\ell, \ell', \rho, \epsilon$ , that uses  $q$  queries, and  $t$  bits of advice, and furthermore  $\epsilon \geq \frac{1}{2^{\ell'/3}}$ ,  $t \leq 2^{\ell/3}$  and  $\rho \geq \frac{1}{2^{\ell/3}}$ , then*

$$q \geq \Omega\left(\frac{1}{\epsilon^\beta}\right) - O(t + \ell).$$

We now explain why Theorem 19 implies the informal statement made in Theorem 9. Recall that in Section 3.1.1 we explained that when using a nonuniform black-box proof to obtain a hard-core predicate, we get a hard-core predicate theorem in which  $s \geq q + t$ .

Theorem 19 implies that for  $s > \ell^{2/\beta}$  it is impossible for such a proof to establish  $\epsilon = 1/s^{2/\beta}$  (even if  $\rho$  is very small). This follows as otherwise, using the fact that  $s \geq q + t \geq t$ , we get that:

$$q \geq \Omega\left(\frac{1}{\epsilon^\beta}\right) - O(t + \ell) \geq \Omega(s^2) - O(t) > s,$$

which is a contradiction to  $s \geq q + t \geq q$ . In particular, the parameter setting considered in Theorem 9, in which  $s = 2^{o(\ell)}$  and  $\epsilon = \frac{1}{s^{\omega(1)}}$ , is impossible to achieve.

## 4 Conclusion and open problems

Unlike Theorem 5 (that handles large  $\epsilon$ ), Theorem 6 (that handles small  $\epsilon$ ) does not achieve a bound of  $q = \Omega\left(\frac{\log(1/\delta)}{\epsilon^2}\right)$ , and only achieves a bound of  $\Omega\left(\frac{1}{\sqrt{\epsilon}}\right)$ . A natural open problem is to improve the bound on  $q$  for small  $\epsilon$  to match the bound for large  $\epsilon$ .

In the case of large  $\epsilon$ , Theorem 5 can be extended to handle local list-decoding from erasures, and gives a lower bound of  $q = \Omega\left(\frac{\log(1/\delta)}{\epsilon}\right)$  on the number of queries of local list-decoders that decode from a  $1 - \epsilon$  fraction of erasures. We do not see how to extend the proof of Theorem 6 to erasures.

The model of black-box proofs that we introduce in Section 3 is quite general, and to the best of our knowledge, covers all known proofs in the literature on hard-core predicates for general one-way functions. Is it possible to circumvent the black-box limitations and answer open problems 8 and 12 for *specific candidates* for one-way functions?

More generally, is it possible to come up with non-black-box techniques that circumvent the limitations?

---

## References

---

- 1 Miklós Ajtai.  $\Sigma_1^1$ -formulae on finite structures. *Annals of Pure and Applied Logic*, 24(1):1–48, 1983. doi:10.1016/0168-0072(83)90038-6.
- 2 Benny Applebaum, Sergei Artemenko, Ronen Shaltiel, and Guang Yang. Incompressible functions, relative-error extractors, and the power of nondeterministic reductions. *Comput. Complex.*, 25(2):349–418, 2016. doi:10.1007/s00037-016-0128-9.
- 3 Sergei Artemenko and Ronen Shaltiel. Lower bounds on the query complexity of non-uniform and adaptive reductions showing hardness amplification. In Leslie Ann Goldberg, Klaus Jansen, R. Ravi, and José D. P. Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques - 14th International Workshop, APPROX 2011, and 15th International Workshop, RANDOM 2011, Princeton, NJ, USA, August 17-19, 2011. Proceedings*, volume 6845 of *Lecture Notes in Computer Science*, pages 377–388. Springer, 2011. doi:10.1007/978-3-642-22935-0\_32.
- 4 Gil Cohen, Anat Ganor, and Ran Raz. Two sides of the coin problem. In Klaus Jansen, José D. P. Rolim, Nikhil R. Devanur, and Cristopher Moore, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2014, September 4-6, 2014, Barcelona, Spain*, volume 28 of *LIPIcs*, pages 618–629. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2014. doi:10.4230/LIPIcs.APPROX-RANDOM.2014.618.
- 5 Yevgeniy Dodis, Abhishek Jain, Tal Moran, and Daniel Wichs. Counterexamples to hardness amplification beyond negligible. In Ronald Cramer, editor, *Theory of Cryptography - 9th Theory of Cryptography Conference, TCC 2012, Taormina, Sicily, Italy, March 19-21, 2012. Proceedings*, volume 7194 of *Lecture Notes in Computer Science*, pages 476–493. Springer, 2012. doi:10.1007/978-3-642-28914-9\_27.
- 6 Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In David S. Johnson, editor, *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washington, USA*, pages 25–32. ACM, 1989. doi:10.1145/73007.73010.
- 7 Aryeh Grinberg, Ronen Shaltiel, and Emanuele Viola. Indistinguishability by adaptive procedures with advice, and lower bounds on hardness amplification proofs. In Mikkel Thorup, editor, *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 956–966. IEEE Computer Society, 2018. doi:10.1109/FOCS.2018.00094.
- 8 Venkatesan Guruswami. Algorithmic results in list decoding. *Foundations and Trends in Theoretical Computer Science*, 2(2), 2006. doi:10.1561/0400000007.
- 9 Dan Gutfreund and Guy N. Rothblum. The complexity of local list decoding. In Ashish Goel, Klaus Jansen, José D. P. Rolim, and Ronitt Rubinfeld, editors, *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques, 11th International Workshop, APPROX 2008, and 12th International Workshop, RANDOM 2008, Boston, MA, USA, August 25-27, 2008. Proceedings*, volume 5171 of *Lecture Notes in Computer Science*, pages 455–468. Springer, 2008. doi:10.1007/978-3-540-85363-3\_36.
- 10 Nutan Limaye, Karteeek Sreenivasaiiah, Srikanth Srinivasan, Utkarsh Tripathi, and S. Venkitesh. A fixed-depth size-hierarchy theorem for  $AC^0[\oplus]$  via the coin problem. In Moses Charikar and Edith Cohen, editors, *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pages 442–453. ACM, 2019. doi:10.1145/3313276.3316339.

- 11 Sofya Raskhodnikova, Noga Ron-Zewi, and Nithin Varma. Erasures versus errors in local decoding and property testing. *Electronic Colloquium on Computational Complexity (ECCC)*, 25:195, 2018. doi:10.4230/LIPIcs.ITCS.2019.63.
- 12 Omer Reingold, Luca Trevisan, and Salil P. Vadhan. Notions of reducibility between cryptographic primitives. In Moni Naor, editor, *Theory of Cryptography, First Theory of Cryptography Conference, TCC 2004, Cambridge, MA, USA, February 19-21, 2004, Proceedings*, volume 2951 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2004. doi:10.1007/978-3-540-24638-1\_1.
- 13 Noga Ron-Zewi, Ronen Shaltiel, and Nithin Varma. Query complexity lower bounds for local list-decoding and hard-core predicates (even for small rate and huge lists). *Electron. Colloquium Comput. Complex.*, 27:133, 2020. URL: <https://eccc.weizmann.ac.il/report/2020/133>.
- 14 Ronen Shaltiel. Is it possible to improve Yao’s XOR lemma using reductions that exploit the efficiency of their oracle? In Jaroslaw Byrka and Raghu Meka, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2020, August 17-19, 2020, Virtual Conference*, volume 176 of *LIPIcs*, pages 10:1–10:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.APPROX/RANDOM.2020.10.
- 15 Ronen Shaltiel and Emanuele Viola. Hardness amplification proofs require majority. *SIAM J. Comput.*, 39(7):3122–3154, 2010. doi:10.1137/080735096.
- 16 Amnon Ta-Shma and David Zuckerman. Extractor codes. *IEEE Trans. Inf. Theory*, 50(12):3015–3025, 2004. doi:10.1109/TIT.2004.838377.
- 17 Emanuele Viola. The complexity of hardness amplification and derandomization, 2006.
- 18 Sergey Yekhanin. Locally decodable codes. *Found. Trends Theor. Comput. Sci.*, 6(3):139–255, 2012. doi:10.1561/04000000030.

# Is the Space Complexity of Planted Clique Recovery the Same as That of Detection?

Jay Mardia

Department of Electrical Engineering, Stanford University, CA, USA  
jmardia@stanford.edu

---

## Abstract

We study the planted clique problem in which a clique of size  $k$  is planted in an Erdős-Rényi graph  $G(n, \frac{1}{2})$ , and one is interested in either detecting or recovering this planted clique. This problem is interesting because it is widely believed to show a statistical-computational gap at clique size  $k = \Theta(\sqrt{n})$ , and has emerged as the prototypical problem with such a gap from which average-case hardness of other statistical problems can be deduced. It also displays a tight computational connection between the detection and recovery variants, unlike other problems of a similar nature. This wide investigation into the computational complexity of the planted clique problem has, however, mostly focused on its time complexity. To begin investigating the robustness of these statistical-computational phenomena to changes in our notion of computational efficiency, we ask-

*Do the statistical-computational phenomena that make the planted clique an interesting problem also hold when we use “space efficiency” as our notion of computational efficiency?*

It is relatively easy to show that a positive answer to this question depends on the existence of a  $O(\log n)$  space algorithm that can recover planted cliques of size  $k = \Omega(\sqrt{n})$ . Our main result comes very close to designing such an algorithm. We show that for  $k = \Omega(\sqrt{n})$ , the recovery problem can be solved in  $O\left(\left(\log^* n - \log^* \frac{k}{\sqrt{n}}\right) \cdot \log n\right)$  bits of space.

1. If  $k = \omega(\sqrt{n} \log^{(\ell)} n)$ <sup>1</sup> for any constant integer  $\ell > 0$ , the space usage is  $O(\log n)$  bits.
2. If  $k = \Theta(\sqrt{n})$ , the space usage is  $O(\log^* n \cdot \log n)$  bits.

Our result suggests that there does exist an  $O(\log n)$  space algorithm to recover cliques of size  $k = \Omega(\sqrt{n})$ , since we come very close to achieving such parameters. This provides evidence that the statistical-computational phenomena that (conjecturally) hold for planted clique time complexity also (conjecturally) hold for space complexity.

**Due to space limitations, we omit proofs from this manuscript. The entire paper with full proofs can be found on arXiv at <https://arxiv.org/abs/2008.12825>.**

**2012 ACM Subject Classification** Theory of computation → Computational complexity and cryptography

**Keywords and phrases** Statistical computational gaps, Planted clique, Space complexity, Average case computational complexity

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.34

**Related Version** arXiv version available at <https://arxiv.org/abs/2008.12825>.

**Funding** This work was supported by a Robert Bosch Stanford Graduate Fellowship.

**Acknowledgements** We would like to thank Dean Doron, Gábor Lugosi, and Kevin Tian for helpful discussions and pointers to relevant literature.

---

<sup>1</sup> Here  $\log^{(\ell)} n$  means we repeatedly take the logarithm of  $n$   $\ell$  times. For example,  $\log^{(3)} n = \log \log \log n$ .



## 1 Introduction

The planted clique problem is a well-studied task in average-case computational complexity, in which a clique of size  $k$  is planted in an Erdős-Rényi graph of size  $n$ ,  $G(n, \frac{1}{2})$ . The problem comes in two flavours, detection ( $\text{PC}_D(n, k)$ ) and recovery ( $\text{PC}_R(n, k)$ ). In the former, we are given either a  $G(n, \frac{1}{2})$  graph or a planted clique graph and must identify the graph we have been given. That is, we must detect whether or not the graph has a planted clique. In the latter, we are given a planted clique graph and must recover all the vertices in the clique.

The planted clique problem shows a variety of interesting phenomena in its time complexity. Not only does it exhibit a statistical-computational gap at clique size  $k = \Theta(\sqrt{n})$ , it has also emerged as the central problem whose average-case hardness implies average-case hardness for many other problems with statistical-computational gaps. See [13, 12] for some examples. Further, the detection and recovery problems have the same threshold at which a polynomial time statistical-computational gap shows up, even though a priori the latter could be a harder problem than the former. In fact, for several other problems such as community detection/recovery in the stochastic block model [1] or planted submatrix detection/recovery [25, 14], there does indeed appear to be a difference between the time complexity of detection and recovery. They become polynomial time feasible at different signal-to-noise ratios, and this makes the lack of a gap between detection and recovery in planted clique all the more noteworthy.

Algorithmic progress on planted cliques has shown that both the detection and recovery problems can be solved “computationally efficiently” (i.e. in polynomial time) for large cliques of size  $k = \Omega(\sqrt{n})$  and less efficiently in quasi-polynomial time  $n^{O(\log n)}$  for cliques larger than the information-theoretic threshold,  $k \geq (2 + \epsilon) \log n$ . The widely believed *Planted Clique Conjecture* even states that if the clique size is small  $k = O(n^{\frac{1}{2}-\delta})$  for any constant  $\delta > 0$ , no polynomial time algorithm can solve the planted clique detection (and hence also the recovery) problem. We survey the results providing evidence for this conjecture in Section 1.1.

However, we do not know how robust these statistical-computational phenomena are to changes in our notion of “computational efficiency”. To begin investigating this, we ask the following question:

*Do the statistical-computational phenomena that make the planted clique an interesting problem also hold when we use “space efficiency” as our notion of computational efficiency?*

To answer this question, we must first discuss what a “space efficient” algorithm is. One of the most well studied classes of space bounded computation is that of logarithmic space, and it is widely considered a benchmark of “space efficient” computation.

Let us further motivate this target space complexity. For deterministic algorithms, this is the class that runs using  $O(\log n)$  bits of space on inputs of size  $\text{poly}(n)$ . It is well known that any deterministic algorithm that uses at most  $s(n)$  bits of space must also run in time  $2^{O(s(n) + \log n)}$  [7, Theorem 4.3]<sup>2</sup>. This means that deterministic logspace algorithms are a

<sup>2</sup> Strictly speaking, the theorem we point to relating deterministic space complexity to time complexity [7, Theorem 4.3] is for Turing machines. While it is convenient to define computational complexity classes using Turing machines, it is extremely inconvenient to design algorithms using them. Instead, we work with a slightly stronger model of computation that allows random access to the input to make algorithm design reasonable. However, the idea behind [7, Theorem 4.3] also holds in any reasonable RAM model and so we ignore this distinction for the purposes of our discussion.

subset of polynomial time algorithms, lending support to the belief that they are a good proxy for “efficient” computation. For algorithms that can use randomness, defining an appropriate notion of space bounded computation involves restricting algorithms that use  $s(n)$  bits of space to at most  $2^{O(s(n)+\log n)}$  time<sup>3</sup>. So, randomized logspace computation corresponds to algorithms running in  $O(\log n)$  space and at most  $\text{poly}(n)$  time on inputs of size  $\text{poly}(n)$ .

This means that if the *Planted Clique Conjecture* is true, no logarithmic space (deterministic or randomized) algorithm can solve the planted clique detection (or recovery) problems for  $k = O(n^{1/2-\delta})$ . If we can show logarithmic space algorithms exist above the polynomial time threshold  $k = \Omega(\sqrt{n})$ , we will have shown that the statistical-computational gap holds even for space complexity.

**Detection.** For detection, essentially the same straightforward algorithms that have been designed for time efficiency can also be implemented space efficiently. For clique sizes above the information theoretic threshold  $k \geq (2 + \epsilon) \log n$ , the same “exhaustively search over sets of  $\Theta(\log n)$  vertices” idea that gives a quasi-polynomial  $n^{O(\log n)}$  time algorithm also gives a  $O(\log^2 n)$  space algorithm<sup>4</sup>. For large cliques above the polynomial time threshold  $k = \Omega(\sqrt{n})$ , the folklore “sum test” or “edge counting” algorithm (see for example Section 1.5 of [36]) can be implemented in  $O(\log n)$  bits of space. We elaborate more on these algorithms in Section 1.3, but for now it suffices to observe that this means a statistical-computational gap holds for planted clique detection at  $k = \Theta(\sqrt{n})$  in terms of space complexity if it holds for time complexity.

**Recovery.** But what about planted clique recovery? Before we go any further, we should clarify what we mean by a small space algorithm for planted clique recovery. The size of the output is  $k \log n$  bits, which could be much larger than the space we are allowing the algorithm. However, the space bound applies only to the working-space of the algorithm, and the output is written on a *write-only* area which does not count towards the space bound. This is standard in the space complexity literature, so we can write-to-output very large answers. See, for example, Section 14.1 of [48].

Just like for detection, simple pre-existing ideas can easily be used to obtain a  $O(\log^2 n)$  space algorithm for recovering planted cliques above the information theoretic threshold, thus matching the detection space complexity in this range of parameters. We provide more details in Section 1.3. Also like for detection, we do not expect a  $O(\log n)$  space algorithm in this regime because of the *Planted Clique Conjecture* and the relation between space and time complexity.

If we can design a  $O(\log n)$  space algorithm that recovers large planted cliques  $k = \Omega(\sqrt{n})$ , we will have shown two things:

- If the conjectured statistical-computational gap at  $k = \Theta(\sqrt{n})$  holds for the time complexity of the planted clique recovery problem, it also holds for space complexity.

<sup>3</sup> See, for example, the section on Randomized Space-Bounded Computation in [7] or the discussion about *BPHSPACE* in [45]. Note that all the algorithms we discuss in this work are deterministic, so we will not need to explicitly analyse or discuss their running time.

<sup>4</sup> Since the best known time complexity for this problem is  $n^{O(\log n)}$ , we do not expect to solve this problem in  $o(\log^2 n)$  bits of space



- Assuming the above statistical-computational gap holds, the coarse-grained<sup>5</sup> computational complexity of planted clique detection and recovery are indeed the same, no matter the notion of complexity we use – time or space.

1. Our first hope for such a logspace recovery algorithm is to see if any pre-existing algorithms are space efficient. However, none of the polynomial time algorithms designed for recovery above  $k = \Omega(\sqrt{n})$  run in small space. They all require at least  $\text{poly}(n)$  bits of space, and in Section 1.3 we discuss, for each of them, why it seems hard to implement them in  $O(\log n)$  bits of space.

Of course, the “degree-counting” polynomial time recovery algorithm for large cliques of size  $k = \Omega(\sqrt{n \log n})$  from [32] *can* easily be implemented in  $O(\log n)$  space. This matches the space complexity for detection in this parameter range. For such large cliques, a simple threshold separates the degrees of non-clique vertices and clique vertices, so membership can easily be decided from a vertex’s degree. A space efficient implementation exists because we can easily count the degree of a vertex (which takes  $O(\log n)$  bits to store) and iterate over all vertices in logarithmic space, re-using the counter used to store the degree across vertices. However, this idea does not work for  $\Omega(\sqrt{n}) = k = o(\sqrt{n \log n})$ , and it is this parameter range in which most algorithmic work for the planted clique problem has been done in the past two decades. If we want to show that the statistical-computational phenomena that hold for time complexity also hold for space complexity, we will need to focus on these parameters.

2. Our next hope is to recall that the lack of a detection-recovery gap in the time complexity of the planted clique problem is not merely an algorithmic coincidence. Section 4.3.3 of [4] shows a black box way to convert a planted clique detection algorithm into a recovery algorithm. The key idea is that if a vertex  $v$  is in the clique, the subgraph induced on the vertex set that *does not* contain  $v$  or its neighbours is distributed as an Erdős-Rényi graph. But, if  $v$  is not in the clique, this induced subgraph is distributed as a planted clique graph. Then we can simply run the detection algorithm to decide if  $v$  is in the planted clique or not<sup>6</sup>. If we could use the edge counting detection algorithm and implement this reduction between recovery and detection in small space, then it seems we would be done. What is more, such a reduction *can* be implemented in small space!<sup>7</sup>

However, there is a slight issue. The statistical success of the reduction in [4] requires the failure probability of the detection algorithm to be at most  $o(\frac{1}{n})$ . This is because we need to repeat the detection algorithm  $n$  times, once for each vertex in the original graph, and thus need to take a union bound. However, as we can see from Section 1.5 in [36], the failure probability of the edge counting test is  $\exp\left(-\Theta\left(\frac{k^4}{n^2}\right)\right)$ . This means the failure probability is  $o(\frac{1}{n})$  only for  $k = \omega(\sqrt{n \log^{\frac{1}{4}} n})$ , which is not a huge improvement over the degree counting algorithm.

<sup>5</sup> By coarse-grained time complexity we mean that we do not distinguish between different  $\text{poly}(n)$  running times. If we were looking at a more fine-grained picture, a gap does emerge between detection and recovery. [37] showed that for  $k = \omega(n^{2/3})$ , planted clique detection can be solved in  $o(n)$  time. However, by results of [41] we know that any recovery algorithm must require  $\Omega(n)$  time. For space complexity, “coarse-grained” means we do not distinguish between any two  $O(\log n)$  space algorithms. Observe that even if there is a  $O(\log n)$  space algorithm that recovers cliques of size  $k = \Omega(\sqrt{n})$ , the fine-grained space complexity of detection and recovery could, in principle, be different. This would be the case if there exists a  $o(\log n)$  space algorithm for detection but not for recovery.

<sup>6</sup> Of course, such a reduction has a built in  $O(n)$  factor time overhead for the recovery algorithm above the detection algorithm.

<sup>7</sup> To count the number of edges induced in such a manner by a vertex  $v$ , we can simply iterate over all pairs  $u, w$  of vertices in the original graph. We increment the counter only if the edge  $(u, w)$  exists *and* neither of the vertices  $u, w$  have an edge to  $v$ .

Due to the discussion above, we need some new ideas to get small space recovery algorithms for planted cliques of size  $k = \Omega(\sqrt{n})$ . Our main result, stated informally below, is one that falls just short of our aim of a  $O(\log n)$  space algorithm. For a formal statement, see Theorem 2.4 in Section 2.

For some large enough constant  $C > 0$ , for planted cliques of size  $k \geq C\sqrt{n}$ , the recovery problem  $\text{PC}_R(n, k)$  can be solved in  $O\left(\left(\log^* n - \log^* \frac{k}{\sqrt{n}}\right) \cdot \log n\right)$  bits of space.

1. If  $k = \omega(\sqrt{n} \log^{(\ell)} n)$  for any constant integer  $\ell > 0$ , the space usage is indeed  $O(\log n)$  bits, which was our target.
2. However, if  $k = C\sqrt{n}$ , the space usage is  $O(\log^* n \cdot \log n)$  bits, which is just shy of what we were aiming for.

Our result suggests that there does exist an  $O(\log n)$  space algorithm to recover cliques of size  $k = \Omega(\sqrt{n})$ , since we come very close to achieving such parameters. We fail to answer our titular question, but only just. We provide strong evidence that the answer is “yes”, and the statistical-computational phenomena that (conjecturally) hold for planted clique time complexity also (conjecturally) hold for space complexity. We have thus initiated the study of high dimensional statistical problems in terms of their space complexity.

As we see in Section 1.1, a long line of work on restricted models of computation has been used to show hardness of the planted clique problem. On the other hand, this work (like [37]) studies a restricted model of computation with the primary aim of making algorithmic progress and further pushing down the complexity of successful planted clique algorithms.

**Open Problem.** Is there a logspace algorithm that recovers planted cliques of size  $k = \Omega(\sqrt{n})$  reliably, or is there a (tiny) detection-recovery gap in the space complexity of the planted clique problem?

## 1.1 Related Work

**Planted Clique Hardness.** It is widely believed that polynomial time algorithms can only detect or recover the planted clique for clique sizes above  $k = \Omega(\sqrt{n})$ . One piece of evidence for this belief is the long line of algorithmic progress using a variety of techniques that has been unable to break this barrier [32, 5, 20, 22, 6, 16, 14, 17, 25, 37]. The other piece of evidence comes from studying restricted but powerful classes of algorithms. [30] showed that a natural Markov chain based technique requires more than polynomial time below this threshold. Similar hardness results (for the planted clique problem or its variants) have been shown for statistical query algorithms [23], circuit classes [43, 44], the Lovász–Schrijver hierarchy [21], and the sum-of-squares hierarchy [39, 18, 26, 9]. Further evidence comes from the low-degree-likelihood method [28, 27, 29, 33] and through concepts from statistical physics [24].

**Statistical-Computational Gaps.** Statistical-computational gaps are not unique to the planted clique problem, and are found in problems involving community detection / recovery [15, 38, 40, 2], sparse PCA [10, 34], tensor PCA [42, 27], random CSPs [3, 31], and robust sparse estimation [35, 8]. However, the planted clique problem is special in that its hardness (or that of its close variants) can be used to show hardness and statistical-computational gaps for a variety of other problems. Such reductions can be seen in [10, 4, 13, 12]. See [12] for a more comprehensive list of examples. To the best of our knowledge, most of these

reductions use randomness quite heavily, so it is unclear if such connections can also be made using only logarithmic space reductions. It would be interesting to do so since this would tie these problems together even more tightly, and would show that planted clique is a central problem in average-case complexity not just for time but also space.

**Detection-Recovery Gaps.** As we have mentioned, the statistical-computational gap in the planted clique problem appears at  $k = \Theta(\sqrt{n})$  for both the detection and recovery variants. This means there is no detection-recovery gap in time complexity, and our work is trying to show that no such gap exists for space complexity either. To understand that the non-existence of this gap is not a foregone conclusion, we note that for several other problems, detection-recovery gaps *do* exist. For example, for communities in the stochastic block model [1], or planted submatrix problems [25, 14]. Moreover, the (non-)existence of a detection-recovery gap is not an inconsequential detail. Since the planted clique problem does not display such a gap, it is not straightforward to use it as a starting point to show detection-recovery gaps for other problems. [12] overcomes this issue for semirandom community recovery by starting from a variant of the planted clique problem, and [46] develops a low-degree-likelihood ratio technique tailored to recovery tasks to get around this problem.

## 1.2 Notation and Problem Definition

**Notation.** We will use standard big  $O$  notation ( $O, \Theta, \Omega$ ). An edge between vertices  $u, v$  is denoted  $(u, v)$ . We let  $\text{Bin}(n, \frac{1}{2})$  denote a Binomial random variable with parameters  $(n, \frac{1}{2})$ . Similarly,  $\text{Bern}(p)$  denotes a Bernoulli random variable that is 1 with probability  $p$  and 0 otherwise. Unless stated otherwise, all logarithms are taken base 2. For a vertex  $v$  in graph  $G = ([n], E)$ , we will denote its degree by  $\deg(v)$ . Throughout this work we identify the vertex set of the graph with the set  $[n] := \{1, 2, \dots, n\}$ . We will also crucially utilise the natural ordering this confers on the names of the vertices.

We also define the so-called binary iterated logarithm  $\log^* n$ .

$$\log^* n = \begin{cases} 0 & \text{if } n \leq 1 \\ 1 + \log^*(\log n) & \text{if } n > 1 \end{cases}$$

Below are formal definitions of the graphs ensembles we use and the planted clique problem.

► **Definition 1.1** (Erdős-Rényi graph distribution:  $G(n, \frac{1}{2})$ ). *Let  $G = ([n], E)$  be a graph with vertex set of size  $n$ . The edge set  $E$  is created by including each possible edge independently with probability  $\frac{1}{2}$ . The distribution on graphs thus formed is denoted  $G(n, \frac{1}{2})$ .*

► **Definition 1.2** (Planted Clique graph distribution:  $G(n, \frac{1}{2}, k)$ ). *Let  $G = ([n], E)$  be a graph with vertex set of size  $n$ . Moreover, let  $K \subset [n]$  be a set of size  $k$  chosen uniformly at random from all  $\binom{n}{k}$  subsets of size  $k$ . For all distinct pairs of vertices  $u, v \in K$ , we add the edge  $(u, v)$  to  $E$ . For all remaining distinct pairs of vertices  $u, v$ , we add the edge  $(u, v)$  to  $E$  independently with probability  $\frac{1}{2}$ . The distribution on graphs thus formed is denoted  $G(n, \frac{1}{2}, k)$ .*

► **Definition 1.3** (Planted Clique Detection Problem:  $\text{PC}_D(n, k)$ ). *This is the following hypothesis testing problem.*

$$H_0 : G \sim G(n, \frac{1}{2}) \quad \text{and} \quad H_1 : G \sim G(n, \frac{1}{2}, k).$$

*Give an algorithm  $\mathcal{A}$  that takes as input the graph  $G$  and outputs either 0 or 1 so that*

$$\Pr(\mathcal{A}(G) = 0 | H_0) + \Pr(\mathcal{A}(G) = 1 | H_1) \geq 4/3.$$

► **Definition 1.4** (Planted Clique Recovery Problem:  $\text{PC}_R(n, k)$ ). *Given an instance of  $G \sim \mathcal{G}(n, \frac{1}{2}, k)$ , recover the planted clique  $K$  with probability at least  $2/3$ .*

### 1.3 Our Techniques

Our space efficient recovery algorithm will depend on the ability to take a small subset of the planted clique and expand it to recover the entire clique. We first discuss such a subroutine, and then talk about our main result, the  $O\left(\left(\log^* n - \log^* \frac{k}{\sqrt{n}}\right) \cdot \log n\right)$  space algorithm for planted clique recovery for large cliques of size  $k = \Omega(\sqrt{n})$ . We do this by first studying polynomial time algorithms that work in this regime, discussing why they take polynomial amounts of space to implement, and then providing the high level ideas of our algorithm. After this, we end with some more details on the straightforward  $O(\log^2 n)$  space implementations of the known quasi-polynomial time algorithms for clique detection and recovery above the information theoretic threshold.

**Small space clique completion.** Several polynomial time recovery algorithms use clique completion / clean-up subroutines to find the entire planted clique after finding just a large enough (possibly noisy) subset of it [5, 22, 16, 17, 37]. However, none of these seem amenable to space efficient implementation, so we create a simple completion algorithm of our own.

We assume we have an algorithm that implicitly maps any planted clique graph to a specific large enough subset of the vertices of the planted clique, which we call  $S_C$ . If given any vertex as input, this algorithm can answer whether this vertex is in  $S_C$  or not using  $s(n)$  bits of space. This is what we mean by “having access to” a subset of the clique that we can now complete. Consider the set  $\tilde{V}$  of those vertices which are connected to every vertex in  $S_C$ . It is easy to show that this new set  $\tilde{V}$  contains the entire planted clique and very few non-clique vertices (see Lemma 3.2). As a result, the number of edges to  $\tilde{V}$  from a clique vertex is far larger than that of a non-clique vertex, and a simple logspace computable threshold can distinguish between the two cases. We show in Algorithm 1 (SMALL SPACE CLIQUE COMPLETION) and Lemma 2.1 that we can use this to decide if a given vertex is in the planted clique or not using  $O(\log n + s(n))$  bits of space. Then we simply loop over all vertices with a further  $O(\log n)$  bits of space and thus have a planted clique recovery algorithm.

**Recovery for  $k = \Omega(\sqrt{n})$ .** We first take a look at existing polynomial time algorithms for  $k = \Omega(\sqrt{n})$  to see why they all require  $\text{poly}(n)$  bits of space, and to see if they have good ideas that we can build on to get small space algorithms.

1. Optimization / SDP algorithms: Several optimization theoretic algorithms involving semidefinite programs have been designed that solve the planted clique recovery problem for  $k = \Omega(\sqrt{n})$  [20, 6, 14, 25]. However, we do not expect to have a general-purpose logarithmic space algorithm for semidefinite programs. The works [19, 47] show that even (approximately) solving linear programs, which are a special case of semidefinite programs, is logspace complete for P. This means that if we had a logspace algorithm for semidefinite programs, every problem with a polynomial time algorithm could also be solved in logarithmic space. Such a proposition is believed to be untrue [48, Conjecture 14.8].
2. (Nearly) Linear time algorithms:
  - a. The algorithm of [22] maintains a subset of “plausible clique vertices” and reduces the size of this subset by 1 in every round. As a result, it needs to maintain a polynomially large subset for most of the time it runs. There also does not seem to be a clever way to compress this set, since it depends crucially on the edge structure of the graph.

- b. The message passing algorithm of [17] is iterative and produces a new dense  $n \times n$  matrix at every iteration, which can not be done in logarithmic space. It is plausible that a more space efficient recursive algorithm that recomputes messages as needed exists. But, since the algorithm requires  $\Theta(\log n)$  sequential iterations / recursive calls, and we will need  $\Omega(\log n)$  bits of space for each level of recursion, we do not expect this space usage to be  $o(\log^2 n)$  bits. Since this does not improve the space usage over the simple algorithm that works above the information theoretic threshold, we do not pursue this idea further.
- c. The algorithm of [16], like [22], maintains a sequence of shrinking subsets of vertices where the ratio between the number of clique and non-clique vertices improves in every round. Further, these subsets are polynomial sized *and* random. Since the pruning of the set depends on randomness from the algorithm, any clever space efficient implementation that re-uses space would need to store the random coins it tosses, defeating the purpose of a space efficient implementation. However, the key idea behind this algorithm *can* be de-randomized, and this is the first observation that forms the basis of our  $O\left(\left(\log^* n - \log^* \frac{k}{\sqrt{n}}\right) \cdot \log n\right)$  space algorithm.

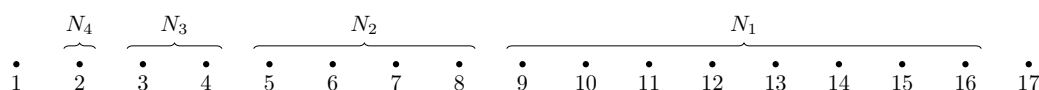
We briefly explain the technique of [16] in more detail, but using the notation of this work rather than that of [16]. Their algorithm runs for  $T$  rounds and maintains a sequence of vertex subsets  $\{N_t, V_t\}_{1 \leq t \leq T}$ .  $N_1 = V_1$  is essentially the entire vertex set  $[n]$ , and then each vertex of  $V_{t-1}$  is included in  $N_t$  iid with some probability and each vertex of  $N_t$  is added to  $V_t$  by cleverly using information from the edge structure of the input graph. This results in the ratio of clique vertices to non-clique vertices in  $V_t$  increasing by a constant factor in every round.  $T$  is then chosen large enough so that  $V_T$  is entirely a subset of the planted clique. The entire clique is now output using a clique completion subroutine.

Since the subsets  $N_t$  described above depend so heavily on the randomness of the algorithm as well as the edge structure of the input graph, this algorithm can not be implemented in less than  $\text{poly}(n)$  space. On the other hand, we have already noted that creating a space efficient clique completion algorithm can be done, and we have done so in Lemma 2.1 with Algorithm 1. So we now focus on trying to modify the first part of the algorithm to something that can be implemented space efficiently. Our challenge is to concisely represent the sets  $N_t$  (and by extension,  $V_t$ ).

Our observation is that the clever filtering of [16] does not depend crucially on the set  $N_t$  being a subset of  $V_{t-1}$  (which is what makes it depend on the edge structure of the graph). Nor does it depend on the set being random. The only thing we really need is that the proportion of clique to non-clique vertices in  $N_t$  is not too small, and that we can easily iterate over all the vertices in any set  $N_t$ . This gives us the freedom we require to design the sets  $N_t$  to be concisely representable, and we use our computer's representation of the vertex set to our advantage. For our computer, the names of the  $n$  vertices of the graph are  $\log n$  bit integers, and we can use the fact that integers have a natural ordering as well as the fact that simple arithmetic can easily be done in  $O(\log n)$  bits of space.

We first set up some notation. The quantities we define will be functions of  $n, k$ , and the graph  $G \sim G(n, \frac{1}{2}, k)$  although our notation will not explicitly denote this. The value of  $n, k$  and the graph will always be clear from context. Recall that  $[n]$  is the vertex set of a graph  $G \sim G(n, \frac{1}{2}, k)$  with  $n$  vertices and a planted clique called  $K$  of size  $k$  and a set of edges called  $E$ .

- Define  $n_0$  as the smallest integer that is a power of 2 and is at least  $n/2$ . This means  $n/2 \leq n_0 < n$ . Define  $k_0 := k \frac{n_0}{n}$
- For any integer  $0 < t < \log n_0$ , let  $n_t := \frac{n_0}{2^t}$ ,  $k_t := \frac{k_0}{2^t}$ . Note that  $n_t$  is always an integer.



■ **Figure 1** An example of our sets  $N_t$  for  $n = 17$ .

- We can now define the subsets  $N_t$  of the vertex set  $[n]$  that will be of particular interest in our filtering algorithm. Let  $N_t := [n_{t-1}] \setminus [n_t]$ , and note that the  $N_t$ 's are all disjoint sets. Clearly,  $|N_t| = n_t$ . See Figure 1 for an example.

It is easy to observe that given  $n, t$  we can iterate over the vertex set  $N_t$  in  $O(\log n)$  space, which is exactly what we wanted. In the analysis of Lemma 2.2, we will also show that the ratio of clique vertices to non-clique vertices in any  $N_t$  is roughly the same as in the whole graph, which is not too small. Now we must implement the rest of the ideas in [16], the ones that actually use the input graph to find the clique.

After setting  $V_1 = N_1$ , the filtering step of [16] fixes a threshold and adds a vertex in  $N_t$  to the set  $V_t$  if and only if that vertex has more edges to  $V_{t-1}$ <sup>8</sup> than the set threshold. Since clique vertices in  $N_t$  are likely to have a higher number of edges to  $V_{t-1}$  than non-clique vertices, the former are more likely to appear in  $V_t$  and the latter are more likely to be filtered out. This is how the ratio of clique to non-clique vertices in  $V_t$  gradually increases with  $t$ . If we had an algorithm to check membership in  $V_{t-1}$  that uses  $s_{t-1}(n)$  bits of space, we could design an algorithm to check for membership in  $V_t$  that uses  $O(\log n) + s_{t-1}(n)$  bits of space. To see this, suppose we have a vertex  $v \in N_t$  and we want to decide if it is in  $V_t$ . We can simply iterate over the set  $N_{t-1}$ , and for each vertex  $u \in N_{t-1}$ , check if it is also in  $V_{t-1}$  using our assumed algorithm. We can also maintain a  $O(\log n)$  bit counter to count the number of edges from  $v$  to all  $u$  that are in  $V_{t-1}$ . Since we can re-use the  $s_{t-1}(n)$  bits of space to check membership in  $V_{t-1}$  across different  $u$ , the whole thing can be done in  $O(\log n) + s_{t-1}(n)$  bits of space. By induction, this means we can check for membership in  $V_T$  using  $O(T \cdot \log n)$  bits of space. We provide a formal algorithm and proof of such a claim in Lemma 2.3 using Algorithm 2.

Overall, this promises to give a  $O(T \cdot \log n)$  space algorithm. What can we set  $T$  to be? Unfortunately, the algorithm of [16] uses  $T = \Theta(\log n)$  rounds, since it only gets a constant factor improvement in the ratio of clique to non-clique vertices in going from  $V_{t-1}$  to  $V_t$ . This gives a  $O(\log^2 n)$  space algorithm, which is not an improvement over the simple algorithm that works above the information theoretic threshold.

Our key idea, inspired by [37], is to implement a better filtering step that gets more than a constant factor of improvement in each round. The filtering / thresholding of [16] does not utilise the size of the planted clique  $k$  at all, other than the fact that it is  $\Omega(\sqrt{n})$ . On the other hand, [37] uses knowledge of  $k$  to design a single round filtering algorithm that recovers the planted clique for clique sizes  $\omega(\sqrt{n} \log \log n) = k = o(\sqrt{n} \log n)$  in sublinear time. By appropriately implementing this idea in our context for multiple rounds, we can utilize knowledge of the number of clique vertices in  $V_{t-1}$ ,  $|V_{t-1} \cap K|$ , to make sure that in going from  $V_{t-1}$  to  $V_t$  the following happens. The number of clique vertices decreases by at most a constant factor, while the number of non-clique vertices decreases by at least a factor of  $\exp\left(\Theta\left(\frac{|V_{t-1} \cap K|^2}{|V_{t-1}|}\right)\right)$ , which is  $\exp\left(\Theta\left(\frac{k^2}{n}\right)\right)$  for  $t = 2$ . For  $k = \Theta(\sqrt{n})$ , this is still a constant factor, but for larger  $k$ , this is much better than a constant factor improvement.

<sup>8</sup> Technically, [16] counts the number of edges to  $V_{t-1} \setminus N_t$ , but in our construction we will have  $V_{t-1} \setminus N_t = V_{t-1}$ .



## 34:10 Is the Space Complexity of Planted Clique Recovery the Same as That of Detection?

To use this idea, our algorithm needs to know  $|V_{t-1} \cap K|$ , which it does not. However, we do have high probability lower bounds on the size  $|V_{t-1} \cap K|$ . We design our thresholds using these estimates, and our analysis in Lemma 2.2 shows that this suffices to get the benefits of this better filter. Let us now define the sets  $V_t$  for our algorithm, thus specifying the filtering threshold. We proceed inductively.

- $V_1 := N_1$
- For any integer  $t > 1$ ,  $V_t$  is a subset of  $N_t$  of vertices which have “large”  $V_{t-1}$ -degree. Quantitatively,  $V_t := \{v \in N_t \mid \sum_{u \in V_{t-1}} \mathbb{1}_{(u,v) \in E} \geq \frac{|V_{t-1}|}{2} + k_{t+2} - 2\sqrt{|V_{t-1}|}\}$ .

It is this carefully chosen threshold sequence which, unlike in [16], varies with  $t$  and uses the value of  $k$  that allows us to improve on the  $O(\log^2 n)$  space bound. In Lemma 2.2 we will show that  $V_T$ , as defined above, is with high probability a subset of the planted clique if  $T$  is large enough. We can implement an algorithm to check membership in  $V_T$  in  $O(T \cdot \log n)$  bits of space as discussed above (and formalized in Lemma 2.3). Moreover, we get the benefits of a very quickly accelerating improvement in the ratio of clique to non-clique vertices from  $V_{t-1}$  to  $V_t$ . From [37] we know that one round of such a filter improves the ratio by a factor of  $\exp\left(\Theta\left(\frac{k^2}{n}\right)\right)$ , and the analysis of our filtering in Lemma 2.2 shows that after  $t$  rounds of such filtering, the ratio improves by what is essentially a tower of exponentials of height  $\frac{t}{2}$ , i.e.  $\exp\left(\exp\left(\dots \exp\left(\Theta\left(\frac{k}{\sqrt{n}}\right)\right)\right)\right)$ . This is why we are able to take  $T = O\left(\log^* n - \log^* \frac{k}{\sqrt{n}}\right)$  (Lemma 2.2). This gives us our main result, an algorithm that can recover planted cliques of size  $k \geq C\sqrt{n}$  in  $O\left(\left(\log^* n - \log^* \frac{k}{\sqrt{n}}\right) \cdot \log n\right)$  bits of space. The formal statement and proof can be found in Theorem 2.4.

### Detection.

1. It is well known (see [11] or Lemma 3.4) that for any positive constant  $\epsilon > 0$ , the probability that an Erdős-Rényi  $G(n, \frac{1}{2})$  graph has a clique of size at least  $(2 + \epsilon) \log n$  goes to 0. Meanwhile, if  $k \geq (2 + \epsilon) \log n$ , then by definition a planted clique graph  $G(n, \frac{1}{2}, k)$  has a clique of size  $(2 + \epsilon) \log n$ . The existence of a clique of this size is a well-known and simple detection test for  $\text{PC}_D(n, k)$  (see, for example, Proposition 1.3 [36]). Moreover, such a test only needs to iterate over all vertex subsets of size  $(2 + \epsilon) \log n$ , which can be done by maintaining a  $\log n$  bit name/number for each of the  $(2 + \epsilon) \log n$  vertices and looping over all possibilities. For a given possible clique, the algorithm needs to check if all  $\binom{(2+\epsilon)\log n}{2}$  edges exist. This can be done by looping over all these edges with 2 more  $O(\log \log n)$  bit counters. Overall, this implementation requires  $O(\log^2 n)$  bits of space.
2. The simple “sum test” or “edge counting” algorithm that is well-known to work for large planted clique  $k = \Omega(\sqrt{n})$  detection (see for example Section 1.5 of [36]) can easily be implemented in  $O(\log n)$  space. The planted graph has significantly more edges than the graph without a clique, so simply counting the number of edges in the input graph and using a threshold test gives a successful detection algorithm. The algorithm only needs to maintain the edge count, which is a number between 1 and  $n^2$  (which can be done with  $O(\log n)$  bits), and it can also easily iterate over all distinct vertex pairs in  $O(\log n)$  bits of space. Lastly, the algorithm also needs to compute the threshold (from [36], we can use the threshold  $\frac{\binom{n}{2}}{2} + \frac{\binom{k}{2}}{4}$ ), which can easily be computed from the input (which contains  $n, k$ ) in logarithmic space. This means that for planted clique detection, assuming we have a time complexity based statistical-computational gap, we also have a space complexity based statistical-computational gap.



**Recovery above the information theoretic threshold.** For cliques of size  $(2 + \epsilon) \log n \leq k = O(\log n)$ , with high probability the planted clique is the unique largest clique in  $\mathcal{G}(n, \frac{1}{2}, k)$  [36, Theorem 1.7]. This means that an algorithm that loops over all possible vertex subsets of size  $k$  can find and output the entire planted clique. To do this it only need to maintain  $k$  names of vertices (which takes  $O(k \log n)$  bits of space) and 2 counters of  $O(\log k)$  bits of space to check if a given set of  $k$  vertices form a clique. Overall, this implementation needs  $O(\log^2 n)$  bits of space.

A simple application of the reduction between detection and recovery from [4] combined with the  $O(\log^2 n)$  space detection algorithm for clique sizes above the information theoretic threshold  $k \geq (2 + \epsilon) \log n$  also gives a  $O(\log^2 n)$  space recovery algorithm for  $k = \omega(\log n)$ . We provide a formal statement and proof in Lemma 3.5.

## 2 Algorithms

We now prove our main results after formalizing our model of computation in Section 2.1. In Section 2.2 we give a space efficient algorithm for clique completion. In Section 2.3 we prove our  $O\left(\left(\log^* n - \log^* \frac{k}{\sqrt{n}}\right) \cdot \log n\right)$  space recovery algorithm for clique sizes above the polynomial time threshold  $k = \Omega(\sqrt{n})$ .

### 2.1 Model of Computation

We use a standard notion of deterministic space bounded computation. See, for example, [48, Section 14.1]. For a  $s(n)$ -space algorithm, the input is a read-only version of the  $n \times n$  adjacency matrix of the graph as well as the clique size  $k$ . Every entry in the matrix as well as the value of  $k$  is stored in its own register. The algorithm has access to  $s(n)$  bits of working space, and the output is write-only (and possibly much larger than  $s(n)$ ). The last fact allows us to solve problems whose outputs may be much larger than  $s(n)$ , a property we will use to solve  $\text{PC}_R(n, k)$ .

To make our model convenient for algorithm design, we also allow random access to the input registers. In our model, we assume basic arithmetic (addition, multiplication, subtraction, division) on  $O(\log n)$  bit numbers can be done in  $O(\log n)$  bits of space. We also assume that the algorithm can compute or knows  $n$  by accessing the adjacency matrix using  $O(\log n)$  bits of space.

### 2.2 Space bounded clique completion

The main idea behind this algorithm is discussed in Section 1.3. If we have access to a large enough subset of the clique, very few vertices that are adjacent to the entire subset (i.e “common neighbours”) are not in the planted clique. Counting the edges from a vertex  $v$  to this set of “common neighbours” of the known clique subset allows us to decide whether or not  $v$  is in the planted clique.

► **Lemma 2.1** (Deterministic + small space clique completion). *Let  $k = \omega(\log n)$ , and  $G \sim \mathcal{G}(n, \frac{1}{2}, k) = ([n], E)$ . Let  $O_{S_C}$  be a deterministic algorithm that uses  $s(n)$  bits of space and, except with probability at most  $p(n) \leq \frac{1}{2}$  (over the randomness in  $G$ ), has the following properties.*

1. *When given as input the graph  $G$  and clique size  $k$ , it implicitly defines a subset of the planted clique vertices  $S_C$  such that  $S_C \subset K$  and  $|S_C| \geq 2 \log n$ .*
2. *It does this by returning, for  $v \in [n]$ ,  $O_{S_C}(v) = 1$  if and only if  $v \in S_C$ , and 0 otherwise.*

---

**Algorithm 1** SMALL SPACE CLIQUE COMPLETION (SSCC).

---

**Input:** Graph  $G = ([n], E) \sim \mathcal{G}(n, \frac{1}{2}, k)$ , clique size  $k$ , oracle  $O_{S_C}$  with access to a clique set  $S_C \subset K : O_{S_C}(v) = 1$  if  $v \in S_C$ ,  $O_{S_C}(v) = 0$  if  $v \notin S_C$

**Output:** Clique  $K$

```

for  $v \in [n]$  do
  Initialize  $\widetilde{\deg}(v) = 0$ 
  for  $u \in [n]$  do
    Initialize  $\text{in}\widetilde{V}(u) = \text{TRUE}$ 
    for  $w \in [n]$  do
      if  $O_{S_C}(w) = 1$  and  $(w, u) \notin E$  then
        Set  $\text{in}\widetilde{V}(u) = \text{FALSE}$ 
      end
    end
    if  $\text{in}\widetilde{V}(u) = \text{TRUE}$  and  $(u, v) \in E$  then
       $\widetilde{\deg}(v) = \widetilde{\deg}(v) + 1$ 
    end
  end
  if  $\widetilde{\deg}(v) \geq \frac{2k}{3} + 3 \log k$  then
    write-to-output  $v$ 
  end

```

} Check if  $u$  is a common neighbour  
 } Use “common neighbour”-degree of  $v$

**end**

---

Then for large enough  $n$ , SMALL SPACE CLIQUE COMPLETION (Algorithm 1), when run on  $G$  with access to the algorithm  $O_{S_C}$ , runs deterministically in space  $O(s(n) + \log n)$  and writes to output the correct planted clique  $K$  except with probability at most  $p(n) + \left(\frac{1}{n}\right)^{\log k} + n \exp\left(-\frac{k}{54}\right)$  (which is over the randomness in  $G$ ).

**Proof.** The proof can be found in the arXiv version at <https://arxiv.org/abs/2008.12825>. ◀

### 2.3 Finding a clique subset in small space

We recall some notation defined in Section 1.3.

- Define  $n_0$  as the smallest integer that is a power of 2 and is at least  $n/2$ . This means  $n/2 \leq n_0 < n$ . Define  $k_0 := k \frac{n_0}{n}$
- For any integer  $0 < t < \log n_0$ , let  $n_t := \frac{n_0}{2^t}$ ,  $k_t := \frac{k_0}{2^t}$ . Note that  $n_t$  is always an integer.
- We also define some subsets of the vertex set  $[n]$  that will be of particular interest in our filtering algorithm. Let  $N_t := [n_{t-1}] \setminus [n_t]$ , and note that the  $N_t$ 's are all disjoint sets. Clearly,  $|N_t| = n_t$ .

So far, we have defined vertex subsets that do not depend at all on the edge structure of the graph. Now we define some subsets that do incorporate information about such edge structure (and hence will be useful in finding the planted clique). We proceed inductively.

- $V_1 := N_1$
- For any integer  $t > 1$ ,  $V_t$  is a subset of  $N_t$ <sup>9</sup> of vertices which have “large”  $V_{t-1}$ -degree<sup>10</sup>.

---

<sup>9</sup> Hence the  $V_t$ 's are all disjoint for different values of  $t$ .

<sup>10</sup> Defined as the number of edges from a vertex  $v \in V_t$  to  $V_{t-1}$ .

Quantitatively,  $V_t := \{v \in N_t \mid \sum_{u \in V_{t-1}} \mathbb{1}_{(u,v) \in E} \geq \frac{|V_{t-1}|}{2} + k_{t+2} - 2\sqrt{|V_{t-1}|}\}$ .

Our main structural lemma shows that for large enough  $T$ ,  $V_T$  is a large enough subset of the planted clique.

► **Lemma 2.2** (Filtering lemma). *Let  $C > 0$  be some large enough constant. Let  $G \sim \mathcal{G}(n, \frac{1}{2}, k)$ , with  $C\sqrt{n} \leq k$  and  $T$  be an integer such that  $2(\log^* n - \log^*(k/\sqrt{n})) + 3 \leq T = O(\log^* n)$ . Then for large enough  $n$ , except with probability at most  $O(\exp(-n^{0.48}))$ ,  $V_T \subset K$  and  $\omega(\log n) = \frac{k}{2^{T+3}} \leq |V_T|$ .*

**Proof.** The proof can be found in the arXiv version at <https://arxiv.org/abs/2008.12825>. ◀

■ **Algorithm 2**  $V_t$ -MEMBERSHIP ( $t \geq 2$ ).

---

**Input:** Graph  $G = ([n], E) \sim \mathcal{G}(n, \frac{1}{2}, k)$ , clique size  $k$ ,  $t$ , vertex  $v \in N_t$ , access to  $V_{t-1}$ -MEMBERSHIP

**Output:** Membership in  $V_t : \mathbb{1}_{v \in V_t}$

Initialize  $\text{size}V_{t-1} = 0, \deg V_{t-1} = 0$

**for**  $u \in N_{t-1}$  **do**

<p><b>if</b> <math>V_{t-1}</math>-MEMBERSHIP(<math>G, k, t-1, u</math>) = 1 <b>then</b></p> <p style="margin-left: 20px;"><math>\text{size}V_{t-1} = \text{size}V_{t-1} + 1</math></p> <p style="margin-left: 20px;"><math>\deg V_{t-1} = \deg V_{t-1} + \mathbb{1}_{(u,v) \in E}</math></p> <p><b>end</b></p>	}	Count $ V_{t-1} $ , “ $V_{t-1}$ ”-degree of $v$
--	---	---

**end**

**output**  $\mathbb{1}_{\{\deg V_{t-1} \geq \frac{\text{size}V_{t-1}}{2} + k_{t+2} - 2\sqrt{\text{size}V_{t-1}}\}}$

---

The  $V_t$ -MEMBERSHIP algorithm simply computes the number of edges from a vertex  $v$  to the set  $V_{t-1}$  and uses this to determine whether or not  $v$  is in  $V_t$ .

► **Lemma 2.3** (Small space filter implementation). *Let  $G = ([n], E) \sim \mathcal{G}(n, \frac{1}{2}, k)$  with a clique size  $k$ . Let  $V_1$ -MEMBERSHIP be an algorithm that returns 1 for every vertex in  $N_1$ , and let  $V_t$ -MEMBERSHIP be defined as in Algorithm 2 for  $t \geq 2$ . Given a vertex  $v \in N_t$ ,  $V_t$ -MEMBERSHIP( $G, k, t, v$ ) returns 1 if and only if  $v \in V_t$ . Otherwise it returns 0. Moreover, it runs in space  $O(t \cdot \log n)$ .*

**Proof.** The proof can be found in the arXiv version at <https://arxiv.org/abs/2008.12825>. ◀

► **Theorem 2.4.** *Let  $G = ([n], E) \sim \mathcal{G}(n, \frac{1}{2}, k)$  with a planted clique of size  $k \geq C\sqrt{n}$  with the constant  $C > 0$  chosen as in Lemma 2.2. Suppose  $T := 2(\log^* n - \log^*(k/\sqrt{n})) + 3$ . Then for large enough  $n$ , there exists a deterministic algorithm that takes as input the adjacency matrix of the graph and the size of the planted clique, exactly outputs the clique  $K$  with probability at least  $1 - O\left(\left(\frac{1}{n}\right)^{\log k}\right)$  over the randomness in the graph  $G$ , and runs using  $O(T \cdot \log n)$  bits of space.*

1. If  $k = C\sqrt{n}$ , the space usage is  $O(\log^* n \cdot \log n)$  bits.
2. If  $k = \omega(\sqrt{n} \log^{(\ell)} n)$  for some constant integer  $\ell > 0$ , the space usage is  $O(\log n)$  bits.

**Proof.** The proof can be found in the arXiv version at <https://arxiv.org/abs/2008.12825>. ◀

### 3 Auxiliary Lemmas

We state the Chernoff bound we use here, for the convenience of the reader.

► **Lemma 3.1.** *Let  $X = \sum_{i=1}^n X_i$  where  $X_i$  are independent  $\text{Bern}(p_i)$  random variables. Let  $\mu = \sum_{i=1}^n p_i$ , and  $0 < \delta$ . Then*

$$\mathbb{P}(X \geq (1 + \delta)\mu) \leq \exp\left(\frac{-\mu\delta^2}{2 + \delta}\right)^{11}$$

$$\mathbb{P}(X \leq (1 - \delta)\mu) \leq \exp\left(\frac{-\mu\delta^2}{3}\right).$$

We state some structural lemmas about the planted clique graph that follow from simple probabilistic arguments.

First we show that with high probability, *any* clique subset of size greater than  $2 \log n$  has at most  $3 \log k$  non-clique vertices connected to every vertex of the subset. The ideas of such an analysis are contained in the proof of [16, Lemma 2.9].

► **Lemma 3.2.** *Let  $G \sim \mathcal{G}(n, \frac{1}{2}, k)$  for  $k \geq 2 \log n$  and  $S$  be any arbitrary subset of the planted clique  $K$  with  $|S| \geq 2 \log n$ . Let  $T$  be the set of all non-clique vertices that are connected to every vertex in  $S$ . Then, except with probability at most  $(\frac{1}{n})^{\log k}$ ,  $|T| \leq 3 \log k$ .*

**Proof.** The proof can be found in the arXiv version at <https://arxiv.org/abs/2008.12825>. ◀

We also control the number of clique vertices any non-clique vertex is connected to.

► **Lemma 3.3.** *Let  $G \sim \mathcal{G}(n, \frac{1}{2}, k)$ , and let  $d$  be the maximum number of clique vertices connected to a non-clique vertex. Then  $\mathbb{P}(d \geq \frac{2k}{3}) \leq n \exp(-\frac{k}{54})$ .*

**Proof.** The proof can be found in the arXiv version at <https://arxiv.org/abs/2008.12825>. ◀

We state the following well known fact that Erdős-Rényi graphs do not have large cliques. See, for example, [11].

► **Lemma 3.4.** *Let  $G \sim \mathcal{G}(n, \frac{1}{2})$  and  $\epsilon > 0$  be a positive constant. Except with probability at most  $O(2^{-\epsilon \log^2 n})$ ,  $G$  contains no cliques of size  $(2 + \epsilon) \log n$  or larger.*

**Proof.** The proof can be found in the arXiv version at <https://arxiv.org/abs/2008.12825>. ◀

We show the existence of a  $O(\log^2 n)$  space recovery algorithm above the information theoretic threshold.

► **Lemma 3.5** ([4] reduction +  $O(\log^2 n)$  space detection). *Let  $\omega(\log n) = k = o(n)$  and  $G \sim \mathcal{G}(n, \frac{1}{2}, k) = ([n], E)$ . Then there is a deterministic  $O(\log^2 n)$  space algorithm that outputs the planted clique except with probability at most  $O(n \exp(-k/54) + n2^{-\Theta(\log^2 n)})$ .*

**Proof.** The proof can be found in the arXiv version at <https://arxiv.org/abs/2008.12825>. ◀

## References

- 1 Emmanuel Abbe. Community detection and stochastic block models: recent developments. *The Journal of Machine Learning Research*, 18(1):6446–6531, 2017.
- 2 Emmanuel Abbe and Colin Sandon. Achieving the  $\kappa$ s threshold in the general stochastic block model with linearized acyclic belief propagation. In *Advances in Neural Information Processing Systems*, pages 1334–1342, 2016.
- 3 Dimitris Achlioptas and Amin Coja-Oghlan. Algorithmic barriers from phase transitions. In *2008 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 793–802. IEEE, 2008.
- 4 Noga Alon, Alexandr Andoni, Tali Kaufman, Kevin Matulef, Ronitt Rubinfeld, and Ning Xie. Testing  $k$ -wise and almost  $k$ -wise independence. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 496–505, 2007.
- 5 Noga Alon, Michael Krivelevich, and Benny Sudakov. Finding a large hidden clique in a random graph. *Random Structures & Algorithms*, 13(3-4):457–466, 1998.
- 6 Brendan PW Ames and Stephen A Vavasis. Nuclear norm minimization for the planted clique and biclique problems. *Mathematical programming*, 129(1):69–89, 2011.
- 7 Sanjeev Arora and Boaz Barak. *Computational complexity: a modern approach*. Cambridge University Press, 2009.
- 8 Sivaraman Balakrishnan, Simon S Du, Jerry Li, and Aarti Singh. Computationally efficient robust sparse estimation in high dimensions. In *Conference on Learning Theory*, pages 169–212, 2017.
- 9 Boaz Barak, Samuel Hopkins, Jonathan Kelner, Pravesh K Kothari, Ankur Moitra, and Aaron Potechin. A nearly tight sum-of-squares lower bound for the planted clique problem. *SIAM Journal on Computing*, 48(2):687–735, 2019.
- 10 Quentin Berthet and Philippe Rigollet. Complexity theoretic lower bounds for sparse principal component detection. In Shai Shalev-Shwartz and Ingo Steinwart, editors, *Proceedings of Machine Learning Research*, volume 30, pages 1046–1066, Princeton, NJ, USA, 12–14 Jun 2013. PMLR. URL: <http://proceedings.mlr.press/v30/Berthet13.html>.
- 11 B Bollobas and P Erdős. Cliques in random graphs. *MPCPS*, 80(3):419, 1976.
- 12 Matthew Brennan and Guy Bresler. Reducibility and statistical-computational gaps from secret leakage. *arXiv preprint*, 2020. [arXiv:2005.08099](https://arxiv.org/abs/2005.08099).
- 13 Matthew Brennan, Guy Bresler, and Wasim Huleihel. Reducibility and computational lower bounds for problems with planted sparse structure. *arXiv preprint*, 2018. [arXiv:1806.07508](https://arxiv.org/abs/1806.07508).
- 14 Yudong Chen and Jiaming Xu. Statistical-computational tradeoffs in planted problems and submatrix localization with a growing number of clusters and submatrices. *arXiv preprint*, 2014. [arXiv:1402.1267](https://arxiv.org/abs/1402.1267).
- 15 Aurelien Decelle, Florent Krzakala, Cristopher Moore, and Lenka Zdeborová. Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications. *Physical Review E*, 84(6):066106, 2011.
- 16 Yael Dekel, Ori Gurel-Gurevich, and Yuval Peres. Finding hidden cliques in linear time with high probability. *Combinatorics, Probability and Computing*, 23(1):29–49, 2014.
- 17 Yash Deshpande and Andrea Montanari. Finding hidden cliques of size  $\sqrt{N/e}$  in nearly linear time. *Foundations of Computational Mathematics*, 15(4):1069–1128, 2015.
- 18 Yash Deshpande and Andrea Montanari. Improved sum-of-squares lower bounds for hidden clique and hidden submatrix problems. In *Conference on Learning Theory*, pages 523–562, 2015.
- 19 David Dobkin, Richard J Lipton, and Steven Reiss. Linear programming is log-space hard for  $p$ . *Information Processing Letters*, 8(2):96–97, 1979.
- 20 Uriel Feige and Robert Krauthgamer. Finding and certifying a large hidden clique in a semirandom graph. *Random Structures & Algorithms*, 16(2):195–208, 2000.
- 21 Uriel Feige and Robert Krauthgamer. The probable value of the lovász–schrijver relaxations for maximum independent set. *SIAM Journal on Computing*, 32(2):345–370, 2003.


- 22 Uriel Feige and Dorit Ron. Finding hidden cliques in linear time. In *21st International Meeting on Probabilistic, Combinatorial, and Asymptotic Methods in the Analysis of Algorithms (AofA'10)*, pages 189–204. Discrete Mathematics and Theoretical Computer Science, 2010.
- 23 Vitaly Feldman, Elena Grigorescu, Lev Reyzin, Santosh S Vempala, and Ying Xiao. Statistical algorithms and a lower bound for detecting planted cliques. *Journal of the ACM (JACM)*, 64(2):1–37, 2017.
- 24 David Gamarnik and Ilias Zadik. The landscape of the planted clique problem: Dense subgraphs and the overlap gap property. *arXiv preprint*, 2019. [arXiv:1904.07174](#).
- 25 Bruce Hajek, Yihong Wu, and Jiaming Xu. Computational lower bounds for community detection on random graphs. In *Conference on Learning Theory*, pages 899–928, 2015.
- 26 Samuel B Hopkins, Pravesh Kothari, Aaron Henry Potechin, Prasad Raghavendra, and Tselil Schramm. On the integrality gap of degree-4 sum of squares for planted clique. *ACM Transactions on Algorithms (TALG)*, 14(3):1–31, 2018.
- 27 Samuel B Hopkins, Pravesh K Kothari, Aaron Potechin, Prasad Raghavendra, Tselil Schramm, and David Steurer. The power of sum-of-squares for detecting hidden structures. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 720–731. IEEE, 2017.
- 28 Samuel B Hopkins and David Steurer. Bayesian estimation from few samples: community detection and related problems. *arXiv preprint*, 2017. [arXiv:1710.00264](#).
- 29 Samuel Brink Klevit Hopkins. *Statistical inference and the sum of squares method*. PhD thesis, Cornell University, 2018.
- 30 Mark Jerrum. Large cliques elude the metropolis process. *Random Structures & Algorithms*, 3(4):347–359, 1992.
- 31 Pravesh K Kothari, Ryuhei Mori, Ryan O'Donnell, and David Witmer. Sum of squares lower bounds for refuting any csp. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 132–145, 2017.
- 32 Luděk Kučera. Expected complexity of graph partitioning problems. *Discrete Applied Mathematics*, 57(2-3):193–212, 1995.
- 33 Dmitriy Kunisky, Alexander S Wein, and Afonso S Bandeira. Notes on computational hardness of hypothesis testing: Predictions using the low-degree likelihood ratio. *arXiv preprint*, 2019. [arXiv:1907.11636](#).
- 34 Thibault Lesieur, Florent Krzakala, and Lenka Zdeborová. Phase transitions in sparse pca. In *2015 IEEE International Symposium on Information Theory (ISIT)*, pages 1635–1639. IEEE, 2015.
- 35 Jerry Li. Robust sparse estimation tasks in high dimensions. *arXiv preprint*, 2017. [arXiv:1702.05860](#).
- 36 Gábor Lugosi. Lectures on combinatorial statistics. *47th Probability Summer School, Saint-Flour*, pages 1–91, 2017.
- 37 Jay Mardia, Hilal Asi, and Kabir Aladin Chandrasekher. Finding planted cliques in sublinear time. *arXiv preprint*, 2020. [arXiv:2004.12002](#).
- 38 Laurent Massoulié. Community detection thresholds and the weak ramanujan property. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 694–703, 2014.
- 39 Raghu Meka, Aaron Potechin, and Avi Wigderson. Sum-of-squares lower bounds for planted clique. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 87–96, 2015.
- 40 Elchanan Mossel, Joe Neeman, and Allan Sly. Consistency thresholds for the planted bisection model. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 69–75, 2015.
- 41 Miklós Z Rácz and Benjamin Schiffer. Finding a planted clique by adaptive probing. *arXiv preprint*, 2019. [arXiv:1903.12050](#).

- 42 Emile Richard and Andrea Montanari. A statistical model for tensor pca. In *Advances in Neural Information Processing Systems*, pages 2897–2905, 2014.
- 43 Benjamin Rossman. On the constant-depth complexity of k-clique. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 721–730, 2008.
- 44 Benjamin Rossman. The monotone complexity of k-clique on random graphs. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 193–201. IEEE, 2010.
- 45 Michael Saks. Randomization and derandomization in space-bounded computation. In *Proceedings of Computational Complexity (Formerly Structure in Complexity Theory)*, pages 128–149. IEEE, 1996.
- 46 Tselil Schramm and Alexander S Wein. Computational barriers to estimation from low-degree polynomials. *arXiv preprint*, 2020. [arXiv:2008.02269](https://arxiv.org/abs/2008.02269).
- 47 Maria Serna. Approximating linear programming is log-space complete for p. *Information Processing Letters*, 37(4):233–236, 1991.
- 48 Avi Wigderson. *Mathematics and Computation: A Theory Revolutionizing Technology and Science*. Princeton University Press, 2019.





# Algorithms and Hardness for Multidimensional Range Updates and Queries

Joshua Lau 

Sydney, Australia

joshua.cs.lau@gmail.com

Angus Ritossa 

University of New South Wales, Sydney, Australia

a.ritossa@unsw.edu.au

---

## Abstract

Traditional orthogonal range problems allow queries over a static set of points, each with some value. Dynamic variants allow points to be added or removed, one at a time. To support more powerful updates, we introduce the GRID RANGE class of data structure problems over arbitrarily large integer arrays in one or more dimensions. These problems allow range updates (such as filling all points in a range with a constant) and queries (such as finding the sum or maximum of values in a range). In this work, we consider these operations along with updates that replace each point in a range with the minimum, maximum, or sum of its existing value, and a constant. In one dimension, it is known that segment trees can be leveraged to facilitate any  $n$  of these operations in  $\tilde{O}(n)$  time overall. Other than a few specific cases, until now, higher dimensional variants have been largely unexplored.

Despite their tightly-knit complexity in one dimension, we show that variants induced by *subsets* of these operations exhibit polynomial separation in two dimensions. In particular, no truly subquadratic time algorithm can support certain pairs of these updates simultaneously without falsifying several popular conjectures. On the positive side, we show that truly subquadratic algorithms can be obtained for variants induced by other subsets.

We provide two general approaches to designing such algorithms that can be generalised to online and higher dimensional settings. First, we give almost-tight  $\tilde{O}(n^{3/2})$  time algorithms for single-update variants where the update and query operations meet a set of natural conditions. Second, for other variants, we provide a general framework for reducing to instances with a special geometry. Using this, we show that  $O(m^{3/2-\epsilon})$  time algorithms for counting paths and walks of length 2 and 3 between vertex pairs in sparse graphs imply truly subquadratic data structures for certain variants; to this end, we give an  $\tilde{O}(m^{(4\omega-1)/(2\omega+1)}) = O(m^{1.478})$  time algorithm for counting simple 3-paths between vertex pairs.

**2012 ACM Subject Classification** Theory of computation → Data structures design and analysis; Mathematics of computing → Graph algorithms

**Keywords and phrases** Orthogonal range, Range updates, Online and Dynamic Data Structures, Fine-grained complexity, Cycle counting

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.35

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/2101.02003>.

**Acknowledgements** We thank Tunan Shi, for suggesting the reduction from 2RANGEINVERSION-QUERY to STATIC TRELLISED 2D GRID RANGE (+, set). We also thank Ray Li and anonymous reviewers for helpful suggestions and comments.

## 1 Introduction

Orthogonal range query problems are ubiquitous across various fields of Computer Science. In the simplest of these problems, a data set is modelled as a set of points in  $\mathbb{Z}^d$ , and the task is to design a data structure that can efficiently answer queries which ask: how many points lie within the (axis-aligned) orthogonal range  $[l_1, r_1] \times \dots \times [l_d, r_d]$ ? One can extend this



© Joshua Lau and Angus Ritossa;

licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 35; pp. 35:1–35:20

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

definition by assigning a value to each point in the input, and having queries ask instead for some aggregate (e.g. the maximum value) of the values of the points within the query range. This has been studied extensively [7, 4, 11, 22] (more in full version), along with models which ask to report all points in the range [2, 3]. *Dynamic* models, where a single point may be inserted or removed in a single operation, have also been studied [13, 12], corresponding to the addition or deletion of a single record. Queries may then be interspersed between these update operations, providing insight into the data set as it changes over time. Modelling data sets in this way has proven useful for Online Analytical Processing (OLAP) [15, 24] of databases.

In practice, however, one may wish to employ more powerful updates. In this work, we examine data structures which support updating the values of all points that fall within a range in addition to range queries. This models updates to the records in a database table whose numerical field values fall within designated ranges. For instance, this could be giving all employees who have been employed between 5 to 10 years, and have KPIs between 80 and 90 an “A” rating. We formalise this as follows.

► **Definition 1.1.** Let  $d$  be a positive integer constant,  $(\mathbb{Z}, q)$  be a commutative semigroup<sup>1</sup> and  $U = \{u_j\}$  be a set of integer functions. The (dD) RANGE  $(q, U)$  problem gives as input a set  $P = \{x_1, \dots, x_p\}$  of  $p$  points in  $\mathbb{Z}^d$ . A corresponding integer value  $v_i$  is also given for each point  $x_i$ , each initially 0. It requests a series of operations, each taking one of the following forms:

- $\text{update}_j((l_1, r_1), \dots, (l_d, r_d))$ : for each  $x_i \in [l_1, r_1] \times \dots \times [l_d, r_d]$ , set  $v_i := u_j(v_i)$
- $\text{query}((l_1, r_1), \dots, (l_d, r_d))$ : compute and return  $q(P')$ , where  $P'$  is the multiset  $\{v_i : x_i \in [l_1, r_1] \times \dots \times [l_d, r_d]\}$ .

Importantly, the operations may include updates of different types, and operations may occur in any order. All operations are given **online** and no information is known about them before the preceding operations are performed. There are  $n = n_u + n_q$  operations in all, with  $n_u$  and  $n_q$  of the first and second forms, respectively.

We are most interested in the case where update functions take the form  $*_c(x) = x * c$ , where  $*$  is a binary operation over the integers, and  $c$  is an operation-specific constant. Through a slight abuse of notation, we also use  $*$  to denote the set of all functions of the form  $*_c$ . We write  $*$  as a member of  $U$  as shorthand for  $*_c$  being a member of  $U$ , for all  $c$ . For example, RANGE  $(+, \{+, \max\})$  allows updates of the form  $+_c$  (increasing values by  $c$ ),  $\max_c$  (replacing values less than  $c$  with  $c$ ), and queries which ask for the sum of values in a range. For a single update function or binary operation  $u$ , we write RANGE  $(q, u)$  for short.

The GRID RANGE variants are those whose point set is  $P = [s]^d$ ;  $P$  is not given explicitly, but rather, is described in the input by the positive integer  $s$ . Hence, the size of input (and thus the running time) can be measured as a function of the number  $n$  of operations which occur, rather than the size of  $P$ . GRID RANGE problems will form our primary focus, but we first describe the context surrounding RANGE problems as a whole.

## 1.1 Motivation

In one dimension, RANGE problems can be viewed as operating over an array. In this case, balanced binary trees (such as binary search trees or *segment trees* [8]) over the array have been effective tools in solving RANGE problems. Such trees allow any range of the array to be canonically expressed as the disjoint union of  $O(\log p)$  subtrees of the tree.

<sup>1</sup> A *semigroup*  $(S, +)$  is a set  $S$  equipped with an associative binary operator  $+: S \times S \rightarrow S$ . A semigroup is *commutative* if  $x + y = y + x$  for all  $x, y \in S$ .

When the functions in  $U$  are closed under composition and each distributes<sup>2</sup> over  $q$ , the folklore technique of *lazy propagation* over a binary tree solves 1D RANGE ( $q, U$ ) in  $O(\log p)$  time in the worst case, per operation. Lazy propagation can be extended to support several types of updates: it can be applied to solve 1D RANGE ( $\max, \{+, \min, \text{set}, \max\}$ ) in worst-case  $O(\log p)$  time per operation, where *set* is the binary operation that returns its second operand. These techniques are described in more detail in Section 2. Ji [18] considered the 1D RANGE ( $+, \max$ ) problem, where the update operation does not distribute over the query operation, by introducing a technique known as a Ji-Driver Tree (informally “segment tree beats”), generalising lazy propagation. Using this technique, he showed that 1D RANGE ( $+, \{+, \min, \text{set}, \max\}$ ) can be solved in amortised  $O(\log^2 p)$  time per operation.

Let  $\mathfrak{B}$  be the set of RANGE ( $q, U$ ) problems with  $q \in \{+, \max\}$  and  $U \subseteq \{+, \min, \text{set}, \max\}$ . Motivated by the one dimensional results for problems in  $\mathfrak{B}$ , we seek general techniques addressing RANGE problems in two or more dimensions. This has been asked as an open question in the competitive programming community [19].

The RANGE problem on a  $p$  element array can be generalised to multiple dimensions in two natural ways: either a set of  $p$  points in  $\mathbb{Z}^d$  is provided explicitly, or the point set is considered to be  $[s]^d$ . In the former case, one dimensional techniques can be generalised to higher dimensions with the aid of an orthogonal space partition tree (hereafter simply *partition tree*), such as a *kd-tree* [7].

► **Lemma 1.2.** *If 1D RANGE ( $q, U$ ) on  $p$  points can be solved in time  $T(p)$  per operation and  $q$  is computable in  $O(1)$  time, then  $d$ D RANGE ( $q, U$ ) can be solved in time  $O(p^{1-1/d}T(p))$  per operation.*

We prove this result and provide almost-tight  $\Omega(p^{1/2-o(1)})$  time per-operation lower bounds conditioned on the Online-Matrix Vector (OMv) Conjecture of Henzinger et al. [16], for all RANGE problems in  $\mathfrak{B}$  when  $d = 2$ , in Section 7.

The latter case corresponds to the GRID RANGE class of problems, which is the subject of the remainder of this work. The same technique does not apply in these cases, as the number of points is  $p = s^d$ .

## 1.2 Prior work

Prior work on GRID RANGE problems has been limited to a few specific cases.

Since balanced binary trees have been rather effective in solving problems in one dimension, it is natural to ask whether they can be easily generalised to higher dimensions. Lueker [21] and Chazelle [13] generalised binary search trees and segment trees to higher dimensions to answer various  $d$ -dimensional range query problems (without updates) on sets of  $n$  points in  $O(\log^d n)$  time per query. This technique can be used to solve GRID RANGE problems in the special cases where all update ranges affect a single point, or when all query ranges contain a single point and the update functions are commutative. The latter can be seen as a generalisation of RECTANGLE STABBING [12] (which accepts a series of  $d$ -dimensional boxes and supports queries for the number of boxes covering a given point), a dual of traditional range queries. The same structure was used with scaling by Ibtehaz et al. [17] to solve GRID RANGE ( $+, +$ ) in worst-case  $O(\log^d s)$  time per operation; without scaling, a straightforward solution for GRID RANGE ( $\max, \max$ ) can be obtained in the same time. These upper bounds contrast with the abovementioned  $\Omega(p^{1/2-o(1)})$  time conditional lower bounds for

<sup>2</sup> An integer function  $u$  distributes over  $q$  if  $u(q(a, b)) = q(u(a), u(b))$

the corresponding 2D RANGE problems. For other problems in  $\mathfrak{B}$ , the lazy propagation technique used in one dimension cannot be directly applied over this structure, as it requires a partition tree of the coordinate space.

In the KLEE'S MEASURE problem,  $n$  rectangles are given in  $d$  dimensions, and one is asked to find the volume of their union. In WEIGHTED DEPTH, the rectangles each have a weight, and one is asked to find the maximum sum of weights of rectangles that cover a single point. In DYNAMIC versions of these problems, a single rectangle can be added or removed in a single operation, and the new answer must be returned after each one. These can be seen as special cases of GRID RANGE problems.

When only additions are supported, DYNAMIC KLEE'S MEASURE is a special case of GRID RANGE (+, set) or GRID RANGE (+, max). Overmars and Yap [23] gave a  $O(n^{(d+1)/2} \log n)$  time solution over  $n$  updates when *the rectangles' coordinates are known during preprocessing*. Chan [9] gave a sublogarithmic time improvement over this method and showed that this is nearly tight in two dimensions, giving an  $\Omega(\sqrt{n})$  time per update worst-case lower bound by reducing from DYNAMIC MATRIX-VECTOR MULTIPLICATION.

DYNAMIC WEIGHTED DEPTH is the special case of GRID RANGE (max, +), where the maximum value in the entire grid is queried after each update. It is closely related to DYNAMIC KLEE'S MEASURE, in that every known algorithm for one has been adaptable to the other, with running times differing by a sublogarithmic factor. Hence, with a slight modification of the result of Overmars and Yap [23], an  $\tilde{O}(n^{(d+1)/2})$  time algorithm for GRID RANGE (max, +) can be obtained. Chan [9] showed that DYNAMIC WEIGHTED DEPTH is solvable in time  $O(n^{(d+1)/2} \log^{d/2} \log n)$ , with sublogarithmic improvements specific to entire-grid queries.

When there are no updates, KLEE'S MEASURE and WEIGHTED DEPTH can be reduced to  $O(n)$  updates on a  $d - 1$  dimensional DYNAMIC instance with a sweepline, however Chan [10] gave faster  $O(n^{d/2-o(1)})$  time algorithms for these. Reductions by Chan [9] and Backurs et al. [6] showed that solving static variants of KLEE'S MEASURE or WEIGHTED DEPTH in time  $o(n^{d\omega/6})$  or  $O(n^{d/2-\epsilon})$  for some  $\epsilon > 0$ , respectively, would improve longstanding upper bounds for CLIQUE or MAX CLIQUE, respectively. It follows that these problems are W[1]-complete for parameter  $d$ . The same reductions can be appropriated for  $d = 3$  to show that an  $O(n^{3/2-\epsilon})$  time algorithm for 3D WEIGHTED DEPTH implies an  $O(n^{3-2\epsilon})$  time algorithm for NEGATIVE TRIANGLE. A truly subcubic algorithm for NEGATIVE TRIANGLE exists if and only if one exists for APSP [26], so an  $O(n^{3/2-\epsilon})$  time algorithm for 2D GRID RANGE (max, +) for some  $\epsilon > 0$  would falsify the APSP Conjecture.

### 1.3 Our contribution

Our main contributions are conditional lower bounds, which serve to elucidate and categorise the expressiveness of these data structures, and upper bounds in the form of general algorithms and frameworks for solving GRID RANGE problems (such as those in  $\mathfrak{B}$ ) in two or more dimensions. We do so on a Word RAM with  $\Omega(\log n)$ -bit words. In the sequel, we use  $\mathfrak{B}'$  to denote  $\mathfrak{B}$  without the problems GRID RANGE (+, +) and GRID RANGE (max, max), for which per-operation polylogarithmic time upper bounds are already known.

**Lower bounds.** First, we consider algorithms whose per-operation time complexity is a function of  $s$ , the side length of the grid. In two dimensions, we show that there is no algorithm running in time  $O(s^{1-\epsilon})$  per-operation for any  $\epsilon > 0$ , for any problem in  $\mathfrak{B}'$ , unless the OMv Conjecture is false. Further, for GRID RANGE (+, {+, max}) we obtain identical lower bounds, conditioned on the “extremely popular conjecture” of Abboud et al. [1] that at

■ **Table 1** Lower and upper bounds for GRID RANGE problems in  $\mathfrak{B}$ , exhibiting polynomial separation. All results are in two dimensions where  $d$  is unspecified, and in  $d$  dimensions otherwise, where  $d$  is a constant. All lower bounds hold offline, except those for OMv, and all upper bounds hold in fully online settings.

$q$	$U$	Lower bounds	Upper bound
max	max		$O(n \log^d s)$ (extension of segment trees)
+	+		$O(n \log^d s)$ [17]
max	+	$\Omega(n^{3/2-o(1)})$ APSP [6] or OMv $\Omega(n^{(d+1)/2-o(1)})$ MAXCLIQUE [6]	$\tilde{O}(n^{3/2})$ and $O(n \log^2 n)$ space $\tilde{O}(n^{(d+1)/2})$ [9, 10]
max	min	$\Omega(n^{3/2-o(1)})$ OMv	$\tilde{O}(n^{(d+1)/2})$
max	set		$\tilde{O}(n^{(d^2+2d-1)/2d})$
max	{min, set, max}		$\tilde{O}(n^{5/4+\omega/(\omega+1)}) = O(n^{1.954})$
+	set		$\tilde{O}(n^{5/4+(4\omega-1)/(4\omega+2)}) = O(n^{1.989})$
+	{+, set}		
max	{+, min}	$\Omega(n^{d-o(1)})$ $d$ -OV	$\tilde{O}(n^d)$
max	{+, min, set, max}		
+	{+, max}	$\Omega(n^{2-o(1)})$ 3SUM $\Omega(n^{d-o(1)})$ $d$ -OV	
+	{+, min, set, max}		

least one of the APSP, 3SUM and 2-OV (Orthogonal Vectors) Conjectures are true (the latter of which is implied by the Strong Exponential Time Hypothesis [27]). Hence, we cannot solve this variant in two dimensions in  $O(s^{1-\epsilon})$  time per operation for any  $\epsilon > 0$  without making a powerful breakthrough across several fields of Theoretical Computer Science. For GRID RANGE (max, {+, min}) and GRID RANGE (+, {+, max}), we generalise our results to  $d$  dimensions under the  $d$ -OV Conjecture to obtain an  $\Omega(s^{(d-1)-o(1)})$  time per-operation lower bound. All these lower bounds are almost-tight, as  $\tilde{O}(s^{d-1})$  time per-operation upper bounds can be obtained by maintaining  $s^{d-1}$  one dimensional instances.

These results, however, do not preclude the existence of efficient and practical algorithms for GRID RANGE problems whose overall complexity is a function of the number of operations,  $n$  – the true size of the input – rather than  $s$ . Our aforementioned lower bounds under the OMv and APSP Conjectures translate to conditional  $\Omega(n^{3/2-o(1)})$  lower bounds for all problems in  $\mathfrak{B}'$ , but our lower bound under the 3SUM Conjecture translates to conditional lower bounds of  $\Omega(n^{2-o(1)})$  for 2D GRID RANGE (+, {+, max}). In  $d$  dimensions, our lower bounds under the  $d$ -OV Conjecture translate to  $\Omega(n^{d-o(1)})$  time conditional lower bounds for GRID RANGE (max, {+, min}) and GRID RANGE (+, {+, max}). Our lower bounds are summarised in Table 1 and proven in Section 3.

**Upper bounds.** By reducing to algorithms in one dimension,  $\tilde{O}(n^d)$  time algorithms can be found for all these problems. Hence, we aim to determine which problems in  $\mathfrak{B}'$  can be solved more efficiently, by seeking truly subquadratic time algorithms in two dimensions. To this end, we provide two general frameworks for developing such algorithms, both based on the approach of Overmars and Yap [23] for DYNAMIC KLEE’S MEASURE. Their algorithm constructs a partition tree of the grid such that the rectangles intersecting each leaf region form a “trellis” pattern. This requires the coordinates of all rectangles to be known during precomputation.

First, we provide a fully-online generalisation of this approach, that does not require coordinates to be known ahead of time.

► **Theorem 1.3.** *Suppose  $q$  and  $u$  are associative, commutative binary operations, computable in  $O(1)$  time, such that  $u$  distributes over  $q$ , and  $0$  is an identity of  $u$ . Then GRID RANGE  $(q, u)$  can be solved in  $\tilde{O}(n^{(d+1)/2})$  time.*

This algorithm does not apply to problems such as the RANGE  $(+, \text{set})$  problem described in the abstract, as set does not distribute over  $+$ . Motivated by this, in Section 5 we show that in two dimensions, efficient solutions to static GRID RANGE  $(q, U)$  instances where the update ranges form the same “trellis” pattern can be used to give fully-online, truly subquadratic time solutions to many GRID RANGE  $(q, U)$  problems. We also extend these results to multiple dimensions, giving a detailed proof in the full version.

As an application, we use this approach to give a truly subquadratic time algorithm for 2D GRID RANGE  $(\max, \{\min, \text{set}, \max\})$ . We do the same for 2D GRID RANGE  $(+, \text{set})$  and 2D GRID RANGE  $(+, \{+, \text{set}\})$  in Section 6, by drawing an equivalence and a reduction between the respective “STATIC TRELLISED” instances and counting the number of 2- and 3-edge paths between vertex pairs, respectively. To this end, we prove the following result.

► **Theorem 1.4.** *Let  $G$  be a graph with  $m$  edges and  $O(m)$  vertices. The number of 3-edge walks between each of  $q$  vertex pairs in  $G$  can be found in  $O(m^{2\omega/(2\omega+1)}(m+q)^{(2\omega-1)/(2\omega+1)})$  time.*

In this way, queries on static graphs yield efficient, fully-online dynamic GRID RANGE data structures. We find it somewhat surprising that, though all of the problems in  $\mathfrak{B}$  can be solved in  $\tilde{O}(n)$  time in one dimension, our upper and lower bounds imply likely polynomial separation in two or more dimensions (see Table 1).

Lastly, we provide a fully-online algorithm for 2D GRID RANGE  $(\max, +)$  that uses  $O(n \log^2 n)$  space, and runs in a time comparable to that of existing algorithms.

► **Theorem 1.5.** *2D GRID RANGE  $(\max, +)$  can be solved in  $\tilde{O}(n^{3/2})$  time, and  $O(n \log^2 n)$  space.*

The proof of this result is omitted for space, and proven in the full version.

While our complexity is slower than existing results by Chan [9] by a polylogarithmic factor, those require  $O(n^{3/2+o(1)})$  space and are not fully-online. Overmars and Yap [23] also gave an  $O(n)$  space algorithm for static KLEE’S MEASURE in  $d$  dimensions using a sweepline, but this does not apply in the dynamic case.

In the remaining sections, due to space constraints, we omit some proofs and sketch others. We refer the reader to the full version for full details and proofs.

## 2 Preliminaries

**Model of computation.** All results are described are for a Word RAM over  $l$ -bit words, with  $l = \Omega(\log n)$ . We further assume that any coordinates or values given in the inputs can be represented in a constant number of words, and that basic arithmetic and the (binary) operations used in range problems can be performed on a constant number of words in constant time. In particular,  $s = O(n^c)$ , for some constant  $c$ .



**Notation.** We use the notation  $\tilde{O}(f(n)) = O(f(n)\text{poly log } n)$  to hide polylogarithmic factors. Note that  $\log^c s = \log^c n$  for any constant  $c$ . Where our algorithms and proofs use positive or negative  $\infty$  as a value, this can be replaced with a suitably large value, for a given input instance.

Where  $x \leq y$  are real numbers, we denote by  $[x, y]$  the set of all *integers* between  $x$  and  $y$ , inclusive. When  $y \geq 1$ , we write  $[y]$  as shorthand for  $[1, y]$ .

The binary operation *set* is the operation whose value is its second operand. That is,  $\text{set}(a, b) = b$ .

$\omega < 2.37286$  [5] is the exponent of multiplying two  $n \times n$  integer matrices. We also write  $\omega(a, b, c)$  for the time taken to multiply an  $a \times b$  matrix by a  $b \times c$  matrix.

**Ancillary problems and variants.** In RANGE problems, we say that the  $i$ th operation (update or query) occurs at *time*  $i$ . In OFFLINE variants, all operations are provided together with the initial input, and in STATIC variants, it is guaranteed that all updates precede all queries.

We formalise and appropriate the “trellis” pattern observed by Overmars and Yap [23] for our use, as follows. Call a GRID RANGE instance TRELLISED if for each update, there is a dimension  $d^*$  such that  $[l_{d'}, r_{d'}] = (-\infty, \infty)$  for all  $d' \in [d] \setminus \{d^*\}$ . When  $d = 2$ , updates must either cover all points in a range of rows, or all points in a range of columns, which we call *row updates* and *column updates*, respectively.

**Segment trees and lazy propagation.** Let  $s$  be a power of two. A *segment tree* over an array  $A$  containing  $s$  elements is a complete rooted binary tree of ranges over  $[s]$ . The root is  $[1, s]$ , and each node  $[a, b]$  has two children:  $[a, h]$ ,  $[h + 1, b]$ , where  $h = (a + b - 1)/2$ . Hence, there are  $O(s)$  nodes in the tree, with a depth of  $\log s$ . Given an interval  $I \subseteq [s]$ , we can write  $I$  as a canonical disjoint union of a set  $\text{base}(I)$  of  $O(\log s)$  nodes. These are defined as the nodes closest to the root that are fully contained in  $I$ , and can be found recursively.

Segment trees can be used to prove the following folklore proposition.

► **Proposition 2.1** (Lazy propagation). *Suppose  $U$  is a set of update functions, and  $q$  is a query function, computable in  $O(1)$  time. If there is a set  $\bar{U}$  such that:*

1.  $U \subseteq \bar{U}$  are sets of functions that can be represented and composed in  $\tilde{O}(1)$  space and time, such that the composition of any series of at most  $n$  (possibly non-distinct) functions of  $U$  results in a function in  $\bar{U}$ ; and
  2. For each  $u \in \bar{U}$ ,  $u$  distributes over  $q$
- then 1D GRID RANGE  $(q, U)$  is solvable in  $\tilde{O}(n)$  time.

**Hardness conjectures.** We base hardness on the following popular conjectures. The first is a conjecture of Henzinger et al. [16].

► **Conjecture 2.2** (OMv Conjecture). *No (randomized) algorithm can process a given  $m \times m$  boolean matrix  $M$ , and then in an online way compute the  $(\vee, \wedge)$ -product  $Mv_i$  for any  $m$  boolean vectors  $v_1, \dots, v_m$  in total time  $O(m^{3-\epsilon})$ , for any  $\epsilon > 0$ .*

In the OuMv problem, the same matrix  $M$  is given during preprocessing, and  $m$  pairs of boolean query vectors  $(u_1, v_1), \dots, (u_m, v_m)$  are given online. For each, the value of the product  $u_i^T Mv_i$  is requested. Note that the answer to each of these queries is a single bit. Henzinger et al. showed that if OuMv can be solved in  $O(m^{3-\epsilon})$  time, then OMv can be solved in  $O(m^{3-\epsilon/2})$  time, so these problems are subcubic equivalent.

We refer the reader to the survey by Vassilevska Williams [28] for more details on the remaining conjectures.

► **Conjecture 2.3** (APSP Conjecture). *No (randomized) algorithm can solve ALL-PAIRS SHORTEST PATHS (APSP) in  $O(v^{3-\epsilon})$  time for  $\epsilon > 0$ , on  $v$  vertex graphs with edge weights in  $\{-v^c, \dots, v^c\}$  and no negative cycles, for large enough  $c$ .*

► **Definition 2.4** ( $k$ -OV problem). *Let  $k \geq 2$  be a constant, and  $z = \omega(\log n)$ . Given  $k$  sets  $A_1, \dots, A_k \subseteq \{0, 1\}^z$  with each  $|A_i| = m$ , determine if there exist  $a_1 \in A_1, \dots, a_k \in A_k$  such that  $a_1 \cdot \dots \cdot a_k = 0$ , where  $a_1 \cdot \dots \cdot a_k := \sum_{i=1}^z \prod_{j=1}^k a_{ji}$ .*

► **Conjecture 2.5** ( $k$ -OV Conjecture). *No (randomized) algorithm can solve  $k$ -OV in  $m^{k-\epsilon} \text{poly}(z)$  time, for any  $\epsilon > 0$ .*

► **Conjecture 2.6** (3SUM Conjecture). *Any algorithm requires  $m^{2-o(1)}$  time in expectation to determine whether a set  $S \subset \{-m^3, \dots, m^3\}$  of  $m$  integers contains three distinct elements  $a, b, c \in S$  with  $a + b = c$ .*

### 3 Conditional lower bounds

In this section, we establish conditional hardness for problems in  $\mathfrak{B}'$  under popular conjectures. We do so by considering per-operation time complexity in terms of  $s$  (the side length of the grid), and overall complexity in terms of  $n$  (the number of operations).

Backurs et al. [6] gave a reduction from MAX  $k$ -CLIQUE to  $k$ D WEIGHTED DEPTH. When  $k = 3$ , MAX  $k$ -CLIQUE is equivalent to NEGATIVE TRIANGLE, which is subcubic equivalent to APSP [26]. Adapting this reduction with a sweepline implies conditional lower bounds for 2D GRID RANGE (max, +).

► **Proposition 3.1** (Modified from [6]). *If OFFLINE 2D GRID RANGE (max, +) can be solved in amortised  $O(s^{1-\epsilon})$  time per update and  $O(s^{2-\epsilon})$  time per query, or in  $O(n^{3/2-\epsilon})$  time overall, for any  $\epsilon > 0$ , then the APSP Conjecture is false.*

We now establish more general linear per-operation lower bounds for 2D GRID RANGE problems in terms of  $s$ , based on the OMv Conjecture.

► **Lemma 3.2.** *Suppose  $(\mathbb{Z}, +, 0)$  is a monoid<sup>3</sup> (resp. group<sup>4</sup>),  $(\mathbb{Z}, \cdot)$  is a commutative semigroup such that  $0r = r0 = 0$  for all  $r \in \mathbb{Z}$  and that there exists  $x \in \mathbb{Z}$  such that  $0 \in \{xz, (x+x)z, (x+x+x)z\}$  if and only if  $z = 0$ . Then, 2D GRID RANGE  $(\cdot, +)$  cannot be solved in worst-case (resp. amortised)  $O(s^{1-\epsilon})$  time per update and  $O(s^{2-\epsilon})$  time per query, for any  $\epsilon > 0$ , unless the OMv Conjecture is false. If  $(\mathbb{Z}, +, 0)$  is a group, 2D GRID RANGE  $(\cdot, +)$  also cannot be solved in  $O(n^{3/2-\epsilon})$  time overall, for any  $\epsilon > 0$ , unless the OMv Conjecture is false.*

**Proof.** We reduce OMv to an instance of 2D GRID RANGE  $(\cdot, +)$  with  $s = m$ . Let  $A_{(i,j)}$  denote the value of the point  $(i, j)$ . Initially, each  $A_{(i,j)} = 0$ . In preprocessing, for each  $M_{ij} = 0$ , add  $x$  to  $A_{(i,j)}$ . We now say the data structure is in its *ready state*.

Let  $(u, v)$  denote a pair of input vectors. For each  $u_i = 0$ , add  $x$  to the value of all points in row  $i$ , and for each  $v_j = 0$ , add  $x$  to the value of all points in column  $j$ . Every point now has a value in  $\{0, x, x+x, x+x+x\}$ . Now some point has value 0 if and only if the answer

<sup>3</sup>  $(\mathbb{Z}, +, 0)$  is a *monoid* if  $(\mathbb{Z}, +)$  is a semigroup, and the identity of  $+$  is 0.

<sup>4</sup>  $(\mathbb{Z}, +, 0)$  is a *group* if it is a monoid and  $+$  is invertible.

to the OuMv query is 1, so we establish this with a single range query. We then restore the data structure to its ready state, either by keeping a journal of updates (semigroup) or by updating with additive inverses (group). The reduction uses  $O(m^2)$  updates and  $O(m)$  queries, implying the stated conditional lower bounds.  $\blacktriangleleft$

This gives a  $\Omega(n^{3/2-o(1)})$  time conditional lower bound on 2D GRID RANGE (max, +), matching that of Proposition 3.1. Through different reductions, we are able to obtain matching lower bounds for the other problems in  $\mathfrak{B}'$ , under the same conjecture.

► **Lemma 3.3.** *If 2D GRID RANGE (+, max), 2D GRID RANGE (+, min) or 2D GRID RANGE (max, min) can be solved in  $O(n^{3/2-\epsilon})$  time overall, for some  $\epsilon > 0$ , then the OMv Conjecture is false.*

The proof is similar to Lemma 3.2, with a different ready state, and is in the full version.

► **Lemma 3.4.** *If 2D GRID RANGE (+, set) or 2D GRID RANGE (max, set) can be solved in  $O(n^{3/2-\epsilon})$  time overall, for some  $\epsilon > 0$ , then the OMv Conjecture is false.*

**Proof sketch.** We reduce from OuMv. In our data structure, we have  $s = m^2$  and utilise an  $m \times m^2$  area of this grid, with updates and queries restricted to this area. In preprocessing, we perform updates so that all points in the  $j$ -th column have a value of  $(j - 1 \bmod m) + 1$ . Each  $M_{ij}$  is represented in  $A$  by a  $1 \times m$  section of points with values  $[1, 2, 3, \dots, m]$ . For each  $M_{ij} = 0$ , we perform an additional update to set its section of points to 0.

For the  $k$ -th query, we only consider columns that were assigned a value of  $m - k + 1$  during preprocessing. Among these, we set to 0 columns  $j$  where  $v_j = 0$ , and query rows  $i$  where  $u_i = 1$  to check for the presence of  $m - k + 1$ .  $\blacktriangleleft$

Together, these give us conditional lower bounds for each of the single-update variants in  $\mathfrak{B}'$ .

► **Corollary 3.5.** *If  $q \in \{+, \max\}$  and  $u \in \{+, \text{set}, \min, \max\}$  and  $q \neq u$ , then 2D GRID RANGE ( $q, u$ ) cannot be solved in worst-case  $O(s^{1-\epsilon})$  time per update and  $O(s^{2-\epsilon})$  time per query, or in  $O(n^{3/2-\epsilon})$  time overall, for some  $\epsilon > 0$ , unless the OMv Conjecture is false. If  $u = +$ , then the lower bounds are amortised rather than worst-case, as addition is invertible.*

When measuring complexity in terms of  $s$ , it appears difficult to improve upon the naive solution which maintains a 1D instance for each column of the grid, for these problems. However, when we measure complexity in terms of  $n$ , there is a polynomial gap between the  $\Omega(n^{3/2-o(1)})$  time lower bound, and the  $\tilde{O}(n^2)$  time naive algorithm. Indeed, Chan [9] gave a  $\tilde{O}(n^{3/2})$  time solution for GRID RANGE (max, +). This might lead one to ask if there exists a general mechanism to adapt  $\tilde{O}(n)$  time algorithms in one dimension to  $\tilde{O}(n^{3/2})$  time algorithms in two dimensions, as there is for RANGE problems on a set of  $n$  explicitly provided points. Alas, when we consider variants with two simultaneous types of updates, we can obtain stronger reductions from the  $d$ -OV and 3SUM Conjectures, suggesting that it is unlikely that such a mechanism exists.

► **Lemma 3.6.** *Let  $d \geq 1$  be a constant. If OFFLINE STATIC TRELLISED  $d$ D GRID RANGE (+, {+, max}) or OFFLINE STATIC TRELLISED  $d$ D GRID RANGE (max, {+, min}) can be solved in amortised  $O(s^{(d-1)-\epsilon})$  time per update and amortised  $O(s^{d-\epsilon})$  time per query, or in  $O(n^{d-\epsilon})$  time overall, for any  $\epsilon > 0$ , then the  $d$ -OV Conjecture is false.*

**Proof sketch.** We reduce from  $d$ -OV with  $z = \log^2 n$  to an instance of  $d$ D GRID RANGE (+, {+, max}) or  $d$ D GRID RANGE (max, {+, min}) with  $s = m$  and  $n = \tilde{O}(m)$ , over the points  $[m]^d$ . Consider the first entry in each vector. Let  $v_{ji}$  be the  $i$ th vector in  $A_j$ . For each  $j \in [d]$

and each vector  $v_{ji} \in A_j$ , using the data structure, add  $(v_{ji})_1$  to all points with  $x_j = i$ . Now a point  $x = (x_1, \dots, x_d) \in [m]^d$  has value  $d$  if and only if  $(v_{jx_j})_1 = 1$  for every  $j \in [d]$ . We then undo these updates by repeating the operations with the negations of the added values, to restore every point to 0. We repeat this procedure for each of the  $z$  entries in the vectors.

Now observe that  $v_{1x_1} \cdot \dots \cdot v_{dx_d} = 0$  if and only if  $C_x$  never attained a value of  $d$  throughout this process. We check if such an  $x$  exists, by adapting a trick of Ji [18] to our data structure. This trick utilises max or min updates, in addition to the  $+$  updates described above.

We note that this reduction can be done offline, uses  $O(mdz) = \tilde{O}(m)$  updates (each of the form  $x_j = i$ ) and a single query spanning the whole grid, giving the required lower bounds.  $\blacktriangleleft$

► **Lemma 3.7.** *If OFFLINE TRELLISED 2D GRID RANGE  $(+, \{+, \max\})$  can be solved with amortised  $O(s^{1-\epsilon})$  time per update and query, or in  $O(n^{2-\epsilon})$  time overall, and any  $\epsilon > 0$ , then the 3SUM Conjecture is false.*

The proof of this result is omitted for space, and proven in the full version.

A modification of Proposition 3.1, together with the results above imply strong conditional hardness for OFFLINE 2D GRID RANGE  $(+, \{+, \max\})$ , when complexity is measured per-operation.

► **Corollary 3.8.** *If OFFLINE 2D GRID RANGE  $(+, \{+, \max\})$  can be solved in amortised  $O(s^{1-\epsilon})$  time per update and query, or in  $O(n^{3/2-\epsilon})$  overall, for any  $\epsilon > 0$ , then the APSP, 2-OV and 3SUM Conjectures are all false.*

Our lower bounds show that a general approach adapting almost-linear one dimensional algorithms for GRID RANGE problems to truly subquadratic solutions for two dimensional instances is unlikely to exist. However, these results do not preclude the existence of efficient and practical algorithms for specific problems, so we would like to classify which 2D GRID RANGE problems can (and can't) be solved in truly subquadratic time. To this end, we now describe truly subquadratic algorithms to some of the as-of-yet unclassified problems in  $\mathfrak{B}'$ , and generalise these to specific subclasses of 2D GRID RANGE problems.

## 4 Solving GRID RANGE problems with a Dynamic Partition

In the full version of the paper, we describe an extension of the partition of Overmars and Yap [23]. Notably, our data structure is fully-online in that it does not require the coordinates in the input to be known during preprocessing. We give a brief overview of the construction and provide some applications, leaving further details to the full version.

### 4.1 Dynamic Structure

First, we introduce some terminology to reason about orthogonal objects in  $d$  dimensions.

A *box* is a  $d$ -dimensional (orthogonal) range. For two boxes  $R_1 = \prod_{i=1}^d [l_i^1, r_i^1]$  and  $R_2 = \prod_{i=1}^d [l_i^2, r_i^2]$ , we say that  $R_1$  is an  *$i$ -pile* with respect to  $R_2$  if  $[l_j^2, r_j^2] \subseteq [l_j^1, r_j^1]$  for all dimensions  $j \in [d] \setminus \{i\}$ . Separately, we say that  $R_1$  *partially covers*  $R_2$  if  $\emptyset \subsetneq R_1 \cap R_2 \subsetneq R_2$ . Similarly,  $R_1$  *completely covers*  $R_2$  if  $R_2 \subseteq R_1$ .

An  *$i$ -slab* ( $i \in [d]$ ) is a box of the form  $[l_1, r_1] \times \dots \times [l_i, r_i] \times \mathbb{Z}^{d-i}$ . We define  $\mathbb{Z}^d$  to be a 0-slab. A *partition tree* is a rooted tree where

1. All nodes are orthogonal ranges of  $\mathbb{Z}^d$
2. The root is  $\mathbb{Z}^d$
3. Every non-leaf node is the disjoint union of its immediate children.

A partition tree is a *level partition tree* if it has depth  $d$ , and the nodes at depth  $i$  ( $i \in [0, d]$ ) are  $i$ -slabs. Hence, a node at depth  $i$  in a level partition tree is *cut* at a series of coordinates in dimension  $i + 1$  to form its children.

► **Theorem 4.1.** *There is a data structure that maintains a set  $V$  of boxes, and supports  $n$  fully-online box insertions to  $V$ . Throughout, it can maintain a level partition tree of  $\mathbb{Z}^d$  whose leaves partition  $\mathbb{Z}^d$  into a set of  $O(n^{d/2})$  axis-aligned regions (colloquially “ $t$ -regions”, short for “trellised-regions”) such that:*

1. *Every box in  $V$  does not intersect, completely covers or is a pile with respect to each  $t$ -region;*
2. *Each box partially covers  $O(n^{(d-1)/2})$   $t$ -regions;*
3. *Each  $t$ -region is partially covered by at most  $O(\sqrt{n})$  boxes;*
4. *Any line parallel to a coordinate axis intersects at most  $O(n^{(d-1)/2})$   $t$ -regions; and*
5. *A list of the boxes that partially cover each  $t$ -region is maintained.*

*This all can be done in amortised  $\tilde{O}(n^{(d-1)/2})$  time per insertion.*

**Proof sketch.** Overmars and Yap [23] construct such a partition tree during precomputation from the boxes’ coordinates, however this is not possible in a fully-online setting. Instead, we maintain the required properties as boxes are added by periodically rebuilding subtrees which exceed a certain size and strategically inserting additional boxes to preserve balance. The cost of rebuilding amortises over the  $n$  operations, giving the stated time complexities. ◀

Using this structure, we can reduce GRID RANGE problems to TRELLISED GRID RANGE problems when the update operation distributes over the query operation.

► **Theorem 4.2.** *Suppose  $q$  and  $u$  are associative operations, both computable in  $O(1)$  time, such that  $q$  is commutative,  $u$  distributes over  $q$ , and  $0$  is an identity of  $u$ . If TRELLISED  $d$ D GRID RANGE  $(q, u)$  can be solved in  $\tilde{O}(n)$  time, then GRID RANGE  $(q, u)$  can be solved in  $\tilde{O}(n^{(d+1)/2})$  time.*

**Proof sketch.** We use the structure given by a dynamic level partition tree  $J$  from Theorem 4.1. Recall that there are  $O(n^{(d-1)/2})$   $(d - 1)$ -slabs in  $J$ , and each is a parent of  $O(\sqrt{n})$   $t$ -regions, which are leaves of  $J$ . For each  $(d - 1)$ -slab, we maintain a data structure supporting range updates and queries within the slab. Conceptually, we maintain a 1D array over its children, in order of their  $d$ -coordinate, with each array entry containing a TRELLISED instance for the corresponding  $t$ -region. We support operations whose ranges completely cover multiple children by using a modified version of the 1D GRID RANGE  $(q, u)$  structure (see Proposition 2.1). Operations affecting only part of a child are handled by the corresponding TRELLISED instance.

Operations are then performed by iterating over all  $(d - 1)$ -slabs, and updating and querying the respective data structures, as necessary. Our data structure is maintained in such a way that over its lifetime, there will be  $O(n^{d/2})$  TRELLISED instances, each facilitating  $O(\sqrt{n})$  operations, giving the required time complexity. ◀

## 4.2 Applications

We apply Theorem 4.2 to give  $\tilde{O}(n^{(d+1)/2})$  time algorithms for particular problems.

First, we show that TRELLISED variants can be solved as separate one dimensional instances, when the conditions of Theorem 4.2 are met and  $u$  is also *commutative*.

► **Lemma 4.3.** *Suppose  $q$  and  $u$  are associative, commutative binary operations, computable in  $O(1)$  time, such that  $u$  distributes over  $q$ , and  $0$  is an identity of  $u$ . Then TRELLISED GRID RANGE  $(q, u)$  is solvable in  $\tilde{O}(n)$  time.*

**Proof.** For notational convenience we provide a proof for the case where  $q$  is  $\max$  and  $u$  is  $+$ ; other operations are proven identically. By the definition of TRELLISED, we can associate every update with a dimension  $i$  such that the update range is an  $i$ -pile with respect to  $\mathbb{Z}^d$ ; we call this an  $i$ -update for short. At any given point in time, let  $U_i(x)$  be the sum (with respect to  $u$ ) of all  $i$ -updates with coordinate  $x$  in dimension  $i$ .

Now consider a query over the range  $R = [l_1, r_1] \times \dots \times [l_d, r_d]$ . The answer to the query can be written as

$$\max_{(x_1, \dots, x_d) \in R} \sum_{i \in [d]} U_i(x_i) = \sum_{i \in [d]} \max_{(x_1, \dots, x_d) \in R} U_i(x_i) = \sum_{i \in [d]} \max_{x_i \in [l_i, r_i]} U_i(x_i)$$

by distributivity. Hence, we can reduce to  $d$  instances of 1D RANGE  $(q, u)$ , which each can be solved in  $\tilde{O}(n)$  time, by Proposition 2.1. ◀

This proves Theorem 1.3, giving efficient fully-online algorithms in these cases.

► **Corollary 4.4.** *GRID RANGE  $(\max, +)$  and GRID RANGE  $(\max, \min)$  can be solved in  $\tilde{O}(n^{(d+1)/2})$  time.*

There also exists some instances where  $u$  is not commutative for which an  $\tilde{O}(n)$  solution to TRELLISED GRID RANGE  $(q, u)$  exists, giving us algorithms with the same time complexity.

► **Lemma 4.5.** *TRELLISED GRID RANGE  $(\max, \text{set})$  can be solved in  $\tilde{O}(n)$  in  $d$  dimensions. Hence, GRID RANGE  $(\max, \text{set})$  can be solved in  $\tilde{O}(n^{(d+1)/2})$ .*

## 5 Reducing to STATIC TRELLISED instances

The technique from the previous section does not work on all variants. For instance, consider the 2D GRID RANGE  $(\max, \{\min, \max\})$  problem. While the update operations ( $\min$  and  $\max$ ) are individually associative, commutative and distribute over the query operation ( $\max$ ), they do not commute with each other. Given that the order of operations matters greatly, it is difficult to decompose this into separate one dimensional problems.

In this section, we describe a general framework for reducing multidimensional range problems to STATIC TRELLISED instances, using 2D GRID RANGE  $(\max, \{\min, \max\})$  as an example. For simplicity, we first give a reduction for OFFLINE instances. We describe how to generalise this approach to online settings, leaving the proof to the full version.

**General approach.** Our algorithm operates by partitioning operations into chronologically contiguous batches of at most  $k$  operations, for some function  $k$  of  $n$ . Each batch may contain both updates and queries. Let  $B$  be a particular batch, and let  $G_B$  be the state of the grid at the start of  $B$ . We will show how to answer the queries within  $B$ .

The  $\bar{k} = O(k)$  coordinates of  $B$ 's operations partition the grid into a  $\bar{k} \times \bar{k}$  *overlay grid* of *overlay regions*. Any update or query will concern all points that fall within a 2D range of whole overlay regions. Since the update operations  $\min_c$  and  $\max_c$  are monotonically increasing functions for any constant  $c$ , it suffices to know the maximum value within each overlay region according to  $G_B$ , to answer the queries of  $B$ . We use these maximums as initial values for a 2D GRID RANGE  $(\max, \{\min, \max\})$  instance over the overlay grid, which we solve by keeping  $\bar{k}$  1D RANGE instances, one for each column, and facilitating operations in  $\tilde{O}(k)$  time, by iterating over each one.



It remains to find the maximum value within each overlay region, according to  $G_B$ . To do so, consider an alternate partition of the grid into t-regions according to Theorem 4.1. In each t-region  $Y$ , we will form an instance of STATIC TRELLISED 2D GRID RANGE ( $\max, \{\min, \max\}$ ) with  $O(\sqrt{n})$  updates; we describe how to do so below. Then, to find the maximum value within an overlay region  $O$ , we issue queries to the instances corresponding to the t-regions intersecting  $O$ .

**Forming STATIC TRELLISED instances.** By construction, each t-region  $Y$  is affected by up to  $n$  whole updates which completely cover  $Y$ , and up to  $\sqrt{n}$  partial updates which cover all points in a range of rows or range of columns of  $Y$ . We can afford to include each partial update in our instance, but need to find a succinct way to represent whole updates.  $Y$  has at most  $\sqrt{n} + 1$  ranges of time between each of its partial updates, and with the aid of a lazy propagation structure over the t-region tree, we can compress the whole updates occurring during each of these ranges into a single update. Hence, we can form an instance of STATIC TRELLISED 2D GRID RANGE ( $\max, \{\min, \max\}$ ) with  $O(\sqrt{n})$  updates, as required.

We now analyse the time complexity of our approach.

► **Lemma 5.1.** *If there an algorithm for STATIC TRELLISED 2D GRID RANGE ( $\max, \{\min, \max\}$ ) running in  $\tilde{O}(n_u^{c-\gamma}(n_u + n_q)^\gamma)$  time for some  $c \geq 1$  and  $\gamma \in [0, 1]$ , then 2D GRID RANGE ( $\max, \{\min, \max\}$ ) can be solved in  $\tilde{O}(n^{5/4+c/2})$  time.*

**Proof.** We process each of the batches separately, and consider the time taken to answer the queries within a particular batch  $B$ .

Using the methods above, in  $\tilde{O}(n^{3/2})$  time we form an instance of STATIC TRELLISED 2D GRID RANGE ( $\max, \{\min, \max\}$ ) with  $n_u = O(\sqrt{n})$  updates in each of  $O(n)$  t-regions. Next, we bound the number of queries made to such instances within  $B$ . Consider the partition  $\mathcal{P}$  of the grid produced by refining each t-region by the overlay grid. A query to a STATIC TRELLISED instance is made for each region in  $\mathcal{P}$ , and the number of these regions and thus, queries, is  $O((k + \sqrt{n})^2) = O(k^2 + n)$ : this is the number of intersections found when the overlay grid is laid atop the t-regions.

For a given t-region  $Y$ , let the number of queries made to the instance in  $Y$  be  $n_{q_Y}$ . First, consider the regions  $y$  where  $n_{q_y} < \sqrt{n}$ . There are  $O(n)$  regions in total, so we spend  $\tilde{O}(n^{1+c/2})$  time answering queries for these regions.

Now consider the regions  $Y$  where  $n_{q_Y} \geq \sqrt{n}$ . Since  $n_{q_Y} = \Omega(n_u)$ , the running time of  $Y$ 's TRELLISED instance is subadditive with respect to  $n_{q_Y}$ . Thus, the total running time for these regions is maximised when the queries are distributed evenly among as many regions as possible. Subject to the constraint on these regions, this maximum is achieved, within a constant multiplicative factor, when there are at most  $O((k^2 + n)/\sqrt{n})$  regions, each with  $\sqrt{n}$  queries. Hence, in this case, the total running time is bounded by  $\tilde{O}((k^2 + n)n^{(c-1)/2})$ . We thus spend time  $\tilde{O}(n^{3/2} + n^{1+c/2} + k^2 n^{(c-1)/2})$  for each of  $n/k$  batches. For balance, we choose  $k = n^{3/4}$ , giving a running time of  $\tilde{O}(n^{5/4+c/2})$  overall. ◀

It remains to show that we can solve STATIC TRELLISED GRID RANGE ( $\max, \{\min, \max\}$ ) efficiently; we do so in the full version. By observing that  $\text{set}_z = \min_z \circ \max_z$ , we obtain the following result.

► **Corollary 5.2.** *2D GRID RANGE ( $\max, \{\min, \text{set}, \max\}$ ) can be solved in  $\tilde{O}(n^{7/4})$  time.*

Most of the steps in our approach are not specific to the update and query operations in our example. To generalise this approach to fully-online and multidimensional settings, we introduce the following “partial information” variant of 1D RANGE.



► **Definition 5.3** (1D PARTITIONED RANGE  $(q, U)$ ). Let  $A_0$  be an integer array of length  $s$  and let  $0 = a_0 < \dots < a_\rho = s$  be a sequence of indices. Let  $Q$  be a corresponding sequence of  $\rho$  integers, denoting the values  $q(A_0[a_0 + 1, a_1]), \dots, q(A_0[a_{\rho-1} + 1, a_\rho])$ . Initially,  $\rho = 1$ .

Let  $J$  be a list of range updates applicable to  $A_0$  in chronological order, initially empty. We write  $A_J$  for the result of applying the updates of  $J$  to  $A_0$ , in order.

Given an integer  $s$ , and the value of  $q(A_0[1, s])$ , support the following operations:

- *split* $(i, a, q_l, q_r)$ : given  $i \in [0, \rho - 1]$  and  $a_i < a < a_{i+1}$ , add  $a$  to the sequence of indices, and update  $Q$  with the knowledge that  $q(A_0[a_i + 1, a]) = q_l$  and  $q(A_0[a + 1, a_{i+1}]) = q_r$
- *update<sub>j</sub>* $(a_l, a_r)$ : append to  $J$ , the update: “for each  $i \in [a_l + 1, a_r]$ , set  $A[i] := u_j(A[i])$ ”
- *query* $(a_l, a_r)$ : return  $q(A_J[a_l + 1, a_r])$

It is guaranteed that every  $a_l$  or  $a_r$  provided as input will already be in the sequence.

Note that this problem is not solvable for all choices of  $q$  and  $U$ : one may need to know the individual values of  $A_0$ , and not just the result of  $q$  over some ranges, to facilitate queries after certain types of updates.

We use this to obtain the following general result, which we prove in the full version.

► **Theorem 5.4.** Suppose  $U$  is a set of update functions, and  $q$  is a query function, computable in  $O(1)$  time. If there is a set  $\bar{U}$  such that:

1.  $U \subseteq \bar{U}$  are sets of functions that can be represented and composed in  $\tilde{O}(1)$  space and time, such that the composition of any series of at most  $n$  (possibly non-distinct) functions of  $U$  results in a function in  $\bar{U}$ ;
2. There is an algorithm for 1D PARTITIONED RANGE  $(q, U)$  that performs both updates and queries in  $\tilde{O}(1)$  time;
3. There is an algorithm for STATIC TRELLISED  $d$ D GRID RANGE  $(q, \bar{U})$  that runs in  $\tilde{O}(n_u^{c-\gamma}(n_u + n_q)^\gamma)$  time for some  $\gamma \in [0, 1]$

then GRID RANGE  $(q, U)$  can be solved in  $\tilde{O}(n^{\frac{c+d+1}{2} - \frac{1}{2d}})$  time. When  $d = 2$ , this is  $\tilde{O}(n^{5/4+c/2})$  time.

This gives a  $\tilde{O}(n^{(d^2+2d-1)/2d})$  time algorithm for GRID RANGE  $(\max, \{\min, \text{set}, \max\})$ .

In the next section, we give two additional applications of this theorem.

## 6 Truly subquadratic set updates and + queries by counting paths

In this section, we apply Theorem 5.4 to give truly subquadratic algorithms for 2D GRID RANGE  $(+, \text{set})$  and 2D GRID RANGE  $(+, \{+, \text{set}\})$ .

### 6.1 2D GRID RANGE $(+, \text{set})$ by counting inversions

The first condition of Theorem 5.4 is met for  $U = \bar{U} = \{\text{set}\}$ , since  $\text{set}_a \circ \text{set}_b = \text{set}_a$  for any  $a$  and  $b$ . The second condition can be met with a data structure for 1D GRID RANGE  $(+, \text{set})$ , maintaining the invariant that a sum query over a range  $R$  in this structure yields the same result as a sum query over  $R$  in the PARTITIONED RANGE structure. Operations each occur in  $O(\log n) = \tilde{O}(1)$  time.

Finally, we address the third condition by drawing an equivalence between STATIC TRELLISED 2D GRID RANGE  $(+, \text{set})$  and a class of range query problems over arrays. The RANGE EQPAIRS QUERY accepts an array of size  $n$  as input, and asks for the number of pairs of equal elements within each of  $q$  given ranges. Duraj et al. [14] defined this weighted analogue for counting inversions between pairs of ranges.

► **Definition 6.1** (WEIGHTED 2RANGEINVERSIONSQUERY). *Given an integer array  $A$ , an integer array of weights  $W$ , both of length  $n$ , and a sequence of  $q$  pairs of non-overlapping ranges  $([l'_1, r'_1], [l''_1, r''_1]), \dots, ([l'_q, r'_q], [l''_q, r''_q])$ , with  $r'_i < l''_i$ , compute for each pair  $([l', r'], [l'', r''])$  the quantity*

$$\sum_{i \in [l', r']} \sum_{j \in [l'', r'']} 1_{A[i] > A[j]} \cdot W[i] \cdot W[j].$$

2RANGEINVERSIONSQUERY is the problem with the added restriction that every weight is 1.

They showed that RANGEQPAIRSQUERY is equivalent, up to polylogarithmic factors, to 2RANGEINVERSIONSQUERY, even when the time complexity is expressed as a function of both  $n$  and  $q$ . We extend this equivalence to STATIC TRELLISED 2D GRID RANGE (+, set).

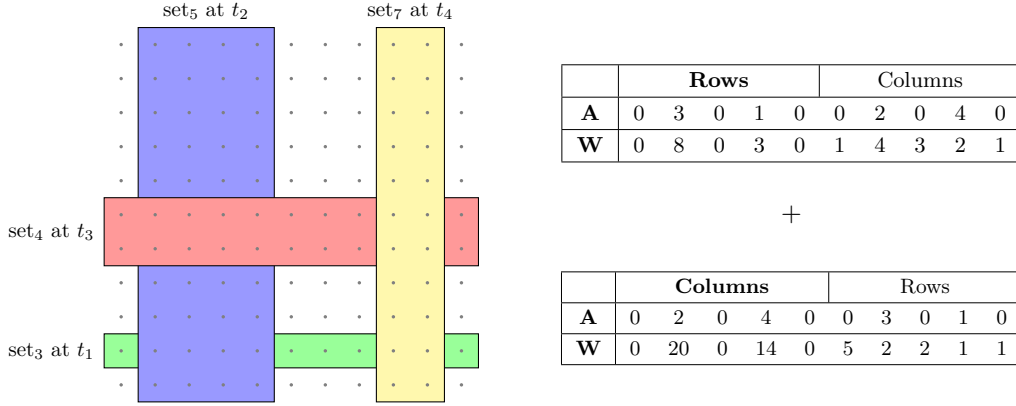
► **Lemma 6.2.** STATIC TRELLISED 2D GRID RANGE (+, set), WEIGHTED 2RANGEINVERSIONSQUERY and 2RANGEINVERSIONSQUERY all have the same complexity, up to polylogarithmic factors. This holds even when the queries are presented online, and with the complexity measured as a function of two variables,  $n$  and  $q$ .

**Proof.** (STATIC TRELLISED 2D GRID RANGE (+, set)  $\rightarrow$  WEIGHTED 2RANGEINVERSIONSQUERY). By adding a dummy update with value 0 covering the whole grid at time  $-1$ , we may assume – without loss of generality – that every row (column) is covered by at least one row (column) update. Now the value of a given point (after all updates) is equal to the value of the later of the latest row update and the latest column update affecting this point's row and column, respectively. Thus, for each row (column), it suffices to keep the latest row (column) update covering it. With a sort and sweep, we can process the row updates into  $O(n)$  disjoint row updates which preserve this property. Hence, without loss of generality, we may assume that the row updates are pairwise disjoint, as are the column updates. This preprocessing takes  $O(n \log n)$  time.

Consider a query over the points  $[l_1, r_1] \times [l_2, r_2]$ . We will show how to compute the contribution (sum of values) of those points whose value is determined by a row update: we can treat the contribution from column updates identically, and the sum of these contributions is the answer to the query. To form our instance of WEIGHTED 2RANGEINVERSIONSQUERY, create an array  $A$ , whose values are equal to the update time of each row update in order, from top to bottom, concatenated with the update time of each column update in order, from left to right. For the weights of our instance, let the weight corresponding to the row update with value  $v$  over rows  $[l_1, r_1]$  be  $v \times (r_1 - l_1 + 1)$ , and the weight corresponding to a column update over columns  $[l_2, r_2]$  be  $(r_2 - l_2 + 1)$ , irrespective of its value. This describes our instance (see Figure 1): we will now describe the queries made to this instance.

Since row (column) updates are disjoint, there is a contiguous range of row (column) updates in this array which lie entirely inside  $[l_1, r_1]$  ( $[l_2, r_2]$ ), whose indices can be found with binary search. The contribution of row updates to the points from these ranges is the result of a query on our instance over the corresponding ranges.  $O(1)$  parts of the query range may fall within part of a row or column update: the contribution from these can be found by scaling the result of a similarly constructed query.

For (WEIGHTED 2RANGEINVERSIONSQUERY  $\rightarrow$  2RANGEINVERSIONSQUERY) and (2RANGEINVERSIONSQUERY  $\rightarrow$  STATIC TRELLISED 2D GRID RANGE (+, set)) reductions, refer to full version. ◀



■ **Figure 1** Reducing STATIC TRELLISED 2D GRID RANGE (+, set) to WEIGHTED 2RANGEIN-VERSIONSQUERY. Updates occur at times  $t_1$  through  $t_4$ . Separate instances for row and column contributions.

Duraj et al. [14] gave an  $\tilde{O}(n^{(2\omega-2)/(\omega+1)}(n+q)^{2/(\omega+1)})$  time<sup>5</sup> algorithm for RANGEQPAIRSQUERY, so we obtain the following truly subquadratic time algorithm for 2D GRID RANGE (+, set).

► **Theorem 6.3.** 2D GRID RANGE (+, set) can be solved in  $\tilde{O}(n^{5/4+\omega/(\omega+1)}) = \tilde{O}(n^{1.954})$  time.

## 6.2 2D GRID RANGE (+, {+, set}) by counting 3-paths

We will once again employ Theorem 5.4 to give a truly subquadratic time algorithm for 2D GRID RANGE (+, {+, set}). The first two conditions are met with slight modifications to that in the previous section; fulfilling the third condition is the subject of the remainder of this section. We begin by defining the following graph problems.

► **Definition 6.4.** The  $k$ -WALKQUERY (resp. SIMPLE $k$ -PATHQUERY) problem gives, as input, a simple graph with  $m$  edges and  $O(m)$  vertices, and poses  $q$  online queries, each asking for the number of  $k$ -edge walks (resp. simple  $k$ -edge paths) between a given pair of vertices.

Duraj et al. [14] proved an equivalence between RANGEQPAIRSQUERY when  $n = q$ , and counting the number of triangles each edge is contained in, in a  $n$ -edge,  $O(n)$ -vertex graph. When the restriction  $n = q$  is relaxed, an equivalence can be drawn with 2WALKQUERY instead. In the same vein, we reduce STATIC TRELLISED 2D GRID RANGE (+, {+, set}) to 3WALKQUERY via a generalisation of RANGEQPAIRSQUERY.

► **Lemma 6.5.** If 3WALKQUERY can be solved in  $T(m, q)$  time then STATIC TRELLISED 2D GRID RANGE (+, {+, set}) can be solved in  $\tilde{O}(T(n_u, n_q))$  time.

To solve 3WALKQUERY, we generalise the 4-cycle detection and counting algorithms of Yuster and Zwick [29] and Vassilevska Williams et al. [25] to solve 3WALKQUERY. Specifically, we partition vertices into three groups based on their degree, and consider each

<sup>5</sup> The multivariate running time given in [14] is slightly better than this when  $q \leq n$ , but this simplified form suffices our purposes.

of the possible configurations of vertices in the 3-Walk with respect to these groups. For each configuration, we use a combination of rectangular matrix multiplication and enumerating edges, both during precomputation and on-the-fly for each query. Finally, we perform a multivariate analysis of the running time, obtaining the following result.

► **Theorem 6.6.** *3WALKQUERY is equivalent to SIMPLE3PATHQUERY, and both can be solved in  $O(m^{2\omega/(2\omega+1)}(m+q)^{(2\omega-1)/(2\omega+1)})$  time.*

Hence, this yields the following algorithm, by Theorem 5.4.

► **Theorem 6.7.** *2D GRID RANGE  $(+, \{+, \text{set}\})$  can be solved in time  $\tilde{O}(n^{5/4+(4\omega-1)/(4\omega+2)}) = O(n^{1.989})$ .*

## 7 RANGE problems over an explicit point set

► **Lemma 1.2.** *If 1D RANGE  $(q, U)$  on  $p$  points can be solved in time  $T(p)$  per operation and  $q$  is computable in  $O(1)$  time, then  $d$ D RANGE  $(q, U)$  can be solved in time  $O(p^{1-1/d}T(p))$  per operation.*

**Proof.** We construct a  $kd$ -tree [7]  $T$  over the points in  $P$ . There is a single point of  $P$  in each of the leaves of  $T$ : we assign these labels from 1 to  $p$ , according to the order of their appearance in a preorder traversal of  $T$ .

We can represent the points in any orthogonal range as those in the disjoint union of  $O(p^{1-1/d})$  subtrees of  $T$  [20]. The labels of points within these subtrees correspond to disjoint ranges of  $[p]$ . Hence, we can keep an instance of 1D RANGE  $(q, U)$ , and perform updates and queries on the corresponding ranges of labels using this data structure. ◀

► **Theorem 7.1.** *If either 2D RANGE  $(+, +)$  or 2D RANGE  $(\max, \max)$  can be solved in amortised  $O(p^{1/2-\epsilon})$  time per update or query, for any  $\epsilon > 0$ , then the OMv Conjecture is false.*

**Proof sketch.** We use a slight modification of Lemma 3.2, with a point in  $P$  for each  $M_{ij} = 1$ . For each pair of query vectors  $(u, v)$ , we use the update operation to mark row  $i$  for each  $u_i = 1$ . For each  $v_j = 1$ , we perform a query over column  $j$  to check if any points in that column were marked. A trick for 2D RANGE  $(\max, \max)$  allows us to reuse the same instance for all query vector pairs. ◀

► **Theorem 7.2.** *If any of 2D RANGE  $(+, \text{set})$ , 2D RANGE  $(\max, \text{set})$ , 2D RANGE  $(\max, +)$ , 2D RANGE  $(+, \max)$  or 2D RANGE  $(\max, \min)$  can be solved in amortised  $O(p^{1/2-\epsilon})$  time per update and amortised  $O(p^{1-\epsilon})$  time per query, for any  $\epsilon > 0$ , then the OMv Conjecture is false.*

The proof is similar to that of Lemma 3.2, and is omitted.

## 8 Open Problems

We have shown that 2D GRID RANGE  $(\max, \{\min, \text{set}, \max\})$  and 2D GRID RANGE  $(\max, +)$  can both be solved in truly subquadratic time, and found  $\Omega(n^{2-o(1)})$  time conditional lower bounds for 2D GRID RANGE  $(\max, \{+, \min\})$ . We also observe that 2D GRID RANGE  $(\max, \{+, \text{set}\})$  reduces to 2D GRID RANGE  $(\max, \{+, \max\})$ , since  $\text{set}_c = \max_c \circ +_{-\infty}$ . Hence, the remaining maximum query variants in  $\mathfrak{B}$  are each at least as hard as 2D GRID RANGE  $(\max, \{+, \text{set}\})$ .

► **Open Problem 8.1.** *Can (OFFLINE) 2D GRID RANGE (max, {+, set}) be solved in truly subquadratic time?*

Among variants supporting sum queries, we gave  $\Omega(n^{2-o(1)})$  time conditional lower bounds for 2D GRID RANGE (+, {+, max}). Using the identity  $\text{set}_c = \text{max}_c \circ +_{-\infty}$  once again, one can see that this is at least as hard as 2D GRID RANGE (+, {+, set}), which we solved in  $O(n^{1.989})$  time, using Theorem 5.4. Another problem easier than 2D GRID RANGE (+, {+, max}) is simply 2D GRID RANGE (+, max), which does not support + updates. This is also the easiest among the remaining sum query variants in  $\mathfrak{B}$ .

We make several comments regarding the hardness of 2D GRID RANGE (+, max). First, we observe that it is also at least as hard as its set “counterpart”, since any instance of GRID RANGE (+, set) can be simulated with two instances of GRID RANGE (+, max).

► **Lemma 8.2.** *GRID RANGE (+, set) can be solved in the same time as GRID RANGE (+, max).*

We solved 2D GRID RANGE (+, set) in  $O(n^{1.954})$  time using Theorem 5.4. However, it is easy to see that there is no solution to 1D PARTITIONED RANGE (+, max), which is required as a precondition of Theorem 5.4: it is simply not enough to know the sum of a range of points. One might instead determine for each overlay region  $O$ , query range  $R$  and  $\text{max}_c$  update: the number of points in  $O \cap R$  with value at most  $c$  and the sum of points in  $O \cap R$  with value greater than  $c$ . This requires  $O(k^3)$  values returned per batch, limiting precomputation to  $O(n^{3/2-\epsilon})$  time, for some  $\epsilon > 0$ , if a truly subquadratic time algorithm overall is desired. This cannot be achieved with a direct application of Theorem 5.4, since there are  $O(n^{3/2})$  updates across the t-regions.

► **Open Problem 8.3.** *Can (OFFLINE) 2D GRID RANGE (+, max) be solved in truly subquadratic time?*

Finally, our  $\Omega(n^{3/2-o(1)})$  conditional lower bounds do not match the upper bounds we gave for 2D GRID RANGE (+, {set, +}), 2D GRID RANGE (+, set) and 2D GRID RANGE (max, {min, set, max}). We ask if the gap can be closed for these problems to see if there exists an in-between complexity class of 2D GRID RANGE problems. In particular, this would be resolved in the affirmative if Theorem 5.4 is a tight reduction for any of these problems.

► **Open Problem 8.4.** *Are there any 2D GRID RANGE problems solvable in  $O(n^{2-\epsilon})$  time, for some  $\epsilon > 0$ , but require  $\Omega(n^{3/2-o(1)})$  time?*

We have studied just a small subset of RANGE and GRID RANGE problems in this work. Additional update or query operations, such as addition modulo a prime, can also be considered. Many existing variants of range searching (see [4]) can also be adapted to these problem classes. In particular, we have not investigated problems which deal with data points that have a “colour” or “category”, and ask for the number of distinct colours in a range. These may be of particular interest, as set updates could be used to facilitate changing the colour of several data points at the same time.

---

## References

- 1 Amir Abboud, Virginia Vassilevska Williams, and Huacheng Yu. Matching triangles and basing hardness on an extremely popular conjecture. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 41–50. ACM, 2015. doi:10.1145/2746539.2746594.

- 2 Peyman Afshani, Lars Arge, and Kasper Dalgaard Larsen. Orthogonal range reporting in three and higher dimensions. In *50th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2009, October 25-27, 2009, Atlanta, Georgia, USA*, pages 149–158. IEEE Computer Society, 2009. doi:10.1109/FOCS.2009.58.
- 3 Peyman Afshani, Lars Arge, and Kasper Dalgaard Larsen. Orthogonal range reporting: query lower bounds, optimal structures in 3-d, and higher-dimensional improvements. In David G. Kirkpatrick and Joseph S. B. Mitchell, editors, *Proceedings of the 26th ACM Symposium on Computational Geometry, Snowbird, Utah, USA, June 13-16, 2010*, pages 240–246. ACM, 2010. doi:10.1145/1810959.1811001.
- 4 Pankaj K. Agarwal. Range searching. In Jacob E. Goodman and Joseph O'Rourke, editors, *Handbook of Discrete and Computational Geometry, Second Edition*, pages 809–837. Chapman and Hall/CRC, 2004. doi:10.1201/9781420035315.ch36.
- 5 Josh Alman and Virginia Vassilevska Williams. A refined laser method and faster matrix multiplication. *CoRR*, abs/2010.05846, 2020. arXiv:2010.05846.
- 6 Arturs Backurs, Nishanth Dikkala, and Christos Tzamos. Tight hardness results for maximum weight rectangles. In Ioannis Chatzigiannakis, Michael Mitzenmacher, Yuval Rabani, and Davide Sangiorgi, editors, *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, volume 55 of *LIPIcs*, pages 81:1–81:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.ICALP.2016.81.
- 7 Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18(9):509–517, 1975. doi:10.1145/361002.361007.
- 8 Jon Louis Bentley. Solutions to klee's rectangle problems. *Unpublished manuscript*, pages 282–300, 1977.
- 9 Timothy M. Chan. A (slightly) faster algorithm for klee's measure problem. In Monique Teillaud, editor, *Proceedings of the 24th ACM Symposium on Computational Geometry, College Park, MD, USA, June 9-11, 2008*, pages 94–100. ACM, 2008. doi:10.1145/1377676.1377693.
- 10 Timothy M. Chan. Klee's measure problem made easy. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 410–419. IEEE Computer Society, 2013. doi:10.1109/FOCS.2013.51.
- 11 Timothy M. Chan, Kasper Green Larsen, and Mihai Pătraşcu. Orthogonal range searching on the ram, revisited. In Ferran Hurtado and Marc J. van Kreveld, editors, *Proceedings of the 27th ACM Symposium on Computational Geometry, Paris, France, June 13-15, 2011*, pages 1–10. ACM, 2011. doi:10.1145/1998196.1998198.
- 12 Timothy M. Chan, Yakov Nekrich, and Michiel H. M. Smid. Orthogonal range reporting and rectangle stabbing for fat rectangles. In Zachary Friggstad, Jörg-Rüdiger Sack, and Mohammad R. Salavatipour, editors, *Algorithms and Data Structures - 16th International Symposium, WADS 2019, Edmonton, AB, Canada, August 5-7, 2019, Proceedings*, volume 11646 of *Lecture Notes in Computer Science*, pages 283–295. Springer, 2019. doi:10.1007/978-3-030-24766-9\_21.
- 13 Bernard Chazelle. A functional approach to data structures and its use in multidimensional searching. *SIAM J. Comput.*, 17(3):427–462, 1988. doi:10.1137/0217026.
- 14 Lech Duraj, Krzysztof Kleiner, Adam Polak, and Virginia Vassilevska Williams. Equivalences between triangle and range query problems. In Shuchi Chawla, editor, *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 30–47. SIAM, 2020. doi:10.1137/1.9781611975994.3.
- 15 Jim Gray, Adam Bosworth, Andrew Layman, and Hamid Pirahesh. Data cube: A relational aggregation operator generalizing group-by, cross-tab, and sub-total. In Stanley Y. W. Su, editor, *Proceedings of the Twelfth International Conference on Data Engineering, February 26 - March 1, 1996, New Orleans, Louisiana, USA*, pages 152–159. IEEE Computer Society, 1996. doi:10.1109/ICDE.1996.492099.



- 16 Monika Henzinger, Sebastian Krinninger, Danupon Nanongkai, and Thatchaphol Saranurak. Unifying and strengthening hardness for dynamic problems via the online matrix-vector multiplication conjecture. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 21–30. ACM, 2015. doi:10.1145/2746539.2746609.
- 17 Nabil Ibtihaz, M. Kaykobad, and M. Sohel Rahman. Multidimensional segment trees can do range queries and updates in logarithmic time. *CoRR*, abs/1811.01226, 2018. arXiv:1811.01226.
- 18 Ruyi Ji. Interval maximum value operation and historical maximum value problem. *Informatics Olympiad China National Team Candidates Essay Collection*, 2016. Article written in Chinese. The author (Ji) has written a blog post in English, outlining the main techniques: <https://codeforces.com/blog/entry/57319>.
- 19 Tuukka Korhonen. On multidimensional range queries. Technical report, University of Helsinki, 2019.
- 20 D. T. Lee and C. K. Wong. Worst-case analysis for region and partial region searches in multidimensional binary search trees and balanced quad trees. *Acta Informatica*, 9:23–29, 1977. doi:10.1007/BF00263763.
- 21 George S. Lueker. A data structure for orthogonal range queries. In *19th Annual Symposium on Foundations of Computer Science, Ann Arbor, Michigan, USA, 16-18 October 1978*, pages 28–34. IEEE Computer Society, 1978. doi:10.1109/SFCS.1978.1.
- 22 Yuzuru Okajima and Kouichi Maruyama. Faster linear-space orthogonal range searching in arbitrary dimensions. In Ulrik Brandes and David Eppstein, editors, *Proceedings of the Seventeenth Workshop on Algorithm Engineering and Experiments, ALENEX 2015, San Diego, CA, USA, January 5, 2015*, pages 82–93. SIAM, 2015. doi:10.1137/1.9781611973754.8.
- 23 Mark H. Overmars and Chee-Keng Yap. New upper bounds in klee’s measure problem. *SIAM J. Comput.*, 20(6):1034–1045, 1991. doi:10.1137/0220065.
- 24 Chung Keung Poon. Dynamic orthogonal range queries in OLAP. *Theor. Comput. Sci.*, 296(3):487–510, 2003. doi:10.1016/S0304-3975(02)00741-7.
- 25 Virginia Vassilevska Williams, Joshua R. Wang, Richard Ryan Williams, and Huacheng Yu. Finding four-node subgraphs in triangle time. In Piotr Indyk, editor, *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 1671–1680. SIAM, 2015. doi:10.1137/1.9781611973730.111.
- 26 Virginia Vassilevska Williams and Ryan Williams. Subcubic equivalences between path, matrix and triangle problems. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*, pages 645–654. IEEE Computer Society, 2010. doi:10.1109/FOCS.2010.67.
- 27 Ryan Williams. A new algorithm for optimal 2-constraint satisfaction and its implications. *Theor. Comput. Sci.*, 348(2-3):357–365, 2005. doi:10.1016/j.tcs.2005.09.023.
- 28 Virginia Vassilevska Williams. Hardness of easy problems: Basing hardness on popular conjectures such as the strong exponential time hypothesis (invited talk). In Thore Husfeldt and Iyad A. Kanj, editors, *10th International Symposium on Parameterized and Exact Computation, IPEC 2015, September 16-18, 2015, Patras, Greece*, volume 43 of *LIPICs*, pages 17–29. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2015. doi:10.4230/LIPICs.IPEC.2015.17.
- 29 Raphael Yuster and Uri Zwick. Detecting short directed cycles using rectangular matrix multiplication and dynamic programming. In J. Ian Munro, editor, *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2004, New Orleans, Louisiana, USA, January 11-14, 2004*, pages 254–260. SIAM, 2004. URL: <http://dl.acm.org/citation.cfm?id=982792.982828>.



# Two Combinatorial MA-Complete Problems

**Dorit Aharonov**

Hebrew University of Jerusalem, Israel  
doria@cs.huji.ac.il

**Alex B. Grilo**

Sorbonne Université, CNRS, LIP6, Paris, France  
Alex.Bredariol-Grilo@lip6.fr

---

## Abstract

Despite the interest in the complexity class MA, the randomized analog of NP, there are just a few known natural (promise-)MA-complete problems. The first such problem was found by Bravyi and Terhal (SIAM Journal of Computing 2009); this result was then followed by Crosson, Bacon and Brown (PRE 2010) and then by Bravyi (Quantum Information and Computation 2015). Surprisingly, each of these problems is either from or inspired by quantum computation. This fact makes it hard for *classical* complexity theorists to study these problems, and prevents potential progress, e.g., on the important question of derandomizing MA.

In this note we define two new natural combinatorial problems and we prove their MA-completeness. The first problem, that we call approximately-clean approximate-connected-component (ACAC), gets as input a succinctly described graph, some of whose vertices are marked. The problem is to decide whether there is a connected component whose vertices are *all unmarked*, or the graph is *far* from having this property. The second problem, called **SetCSP**, generalizes in a novel way the standard constraint satisfaction problem (CSP) into constraints involving *sets* of strings.

Technically, our proof that **SetCSP** is MA-complete is a fleshing out of an observation made in (Aharonov and Grilo, FOCS 2019), where it was noted that a restricted case of Bravyi and Terhal's MA complete problem (namely, the *uniform* case) is already MA complete; and, moreover, that this restricted case can be stated using classical, combinatorial language. The fact that the first, arguably more natural, problem of ACAC is MA-hard follows quite naturally from this proof as well; while containment of ACAC in MA is simple, based on the theory of random walks.

We notice that this work, along with a translation of the main result of Aharonov and Grilo to the **SetCSP** problem, implies that finding a gap-amplification procedure for **SetCSP** (in the spirit of the gap-amplification procedure introduced in Dinur's PCP proof) would imply  $MA=NP$ . In fact, the problem of finding gap-amplification for **SetCSP** is *equivalent* to the  $MA=NP$  problem. This provides an alternative new path towards the major problem of derandomizing MA. Deriving a similar statement regarding gap amplification of a natural restriction of ACAC remains an open question.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Complexity classes

**Keywords and phrases** Merlin-Arthur proof systems, Constraint satisfaction problem, Random walks

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.36

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/2003.13065>.

**Funding** Dorit Aharonov: ISF grant 1721/17.

**Acknowledgements** We notice that the ACAC problem did not appear in the first version of the this work and we thank an anonymous reviewer who pushed us to define more natural problems. We are grateful to Umesh Vazirani for asking the right question that led to this note. Most of this work was done while A.G. was affiliated to CWI and QuSoft, and part of it was done while A.G. was visiting the Simons Institute for the Theory of Computing.



© Dorit Aharonov and Alex B. Grilo;  
licensed under Creative Commons License CC-BY  
12th Innovations in Theoretical Computer Science Conference (ITCS 2021).  
Editor: James R. Lee; Article No. 36; pp. 36:1–36:20



Leibniz International Proceedings in Informatics  
LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

## 1 Introduction

The complexity class MA is a natural extension of NP proof systems to the probabilistic setting [4]. There is a lot of evidence towards the fact that these two complexity classes are equal [14, 12, 13, 5, 18, 15, 20, 16], however the proof remains elusive. It is even open to show that every problem in MA can be solved in non-deterministic *sub-exponential* time.

Surprisingly, the very first *natural* MA-complete<sup>1</sup> problem, found by Bravyi and Terhal [8] only close to 25 years after the definition of the class (!) is defined using *quantum* terminology. But why would randomized NP have anything to do with quantum? Bravyi and Terhal show that deciding if a given stoquastic  $k$ -Local Hamiltonian<sup>2</sup> is frustration-free or at least inverse polynomially frustrated, is promise-MA-complete. Bravyi [6] also proved MA-completeness of yet another quantum Hamiltonian problem. A third MA-complete problem was proposed by Crosson, Bacon and Brown [9], inspired by quantum adiabatic computation.<sup>3</sup>

This leaves us in a strange situation in which the known MA complete problems are not stated using natural or standard complexity theory terminology. This makes us wonder if there is a fundamental reason why we cannot find classical MA-complete problems, or it is just an “accident” that the first MA-complete problems come from quantum computing. Moreover, we think that new natural (classically defined) complete problems for the class MA might enable access to the major open problem of derandomization of MA, and possibly to other related problems such as PCP for MA [1]<sup>4</sup>. In particular, though the problems proposed in [8, 6] are very natural within quantum complexity theory, the fact that they are defined within the area of quantum computation seems to pose a language and conceptual barrier that might delay progress on them, and make it hard for *classical* complexity theorists to study them.

The goal of this work is to provide classically, combinatorially-defined complete problems for MA. We hope that these definitions lead to further understanding of the class MA and the MA vs. NP question.

One of these problems, **SetCSP**, is morally based on the MA complete problem of [8], while the other problem, **ACAC**, seems to be a rather natural problem on graphs.

The definition of **SetCSP** as well as the proof of its MA-completeness rely on a simple but crucial insight: we can translate “testing history states”, a notion familiar in quantum complexity theory, into constraints on *sets* of strings. This idea is used here throughout, but in fact can be used to translate also other quantum results related to stoquastic Hamiltonians, to the classical combinatorial language of constraints on *sets of strings* (see Section 4.4.1 as

<sup>1</sup> For PromiseMA, it is folklore that one can define complete problems by extending NP-complete problems (see, e.g. [19]): we define an exponential family of 3SAT formulas (given as input succinctly) and we have to decide if there is an assignment that satisfies all of the formulas, or for every assignment, a random formula in the family will not be satisfied with good probability.

<sup>2</sup> Stoquastic Hamiltonians sit between classical Hamiltonians (CSPs) and general quantum Hamiltonians.

<sup>3</sup> Crosson et. al.’s problem asks about the properties of the Gibbs distribution corresponding to the temperature  $T$  of a specific family of *classical*, rather than quantum, physical systems, defined using local classical Hamiltonians. Though inspired by adiabatic quantum computation, this problem is in fact defined using *classical* terminology, since the classical Hamiltonians can be viewed as constraint satisfaction systems. Yet, we note that its definition uses a layer of physical notation, involving Gibbs distribution and temperature. Moreover, when stated using classical terminology, the input to this problem is restricted, in a fairly contrived way, to sets of classical constraints which can be associated with a (noisy) deterministic circuit. Both of these aspects seem to make handling this problem using standard combinatorial tools difficult or at least not very natural.

<sup>4</sup> We notice that Drucker [11] proves a PCP theorem for AM; in the definition he uses for AM, the coins are public and the prover sees them (See Section 2.3 in [11]); but his result does not hold when the coins are private, namely for MA.

well as Figure 1 for more intuition). However though this translation idea can be viewed as standing at the heart of many of the definitions and results presented here, it is in fact *not* necessary to understand the proofs.

Based on this idea, as well as a simple observation made in [1] which says that the problem of [8] remains promise-MA complete even when restricted to what is referred to in [1] as *uniform* stoquastic Hamiltonians, we can prove the MA-hardness proof of the SetCSP problem as a translation of the MA-hardness proof given in [8] into a classical language. We notice that the proof of [8] on its own heavily relies on the Quantum Cook-Levin proof of Kitaev [17].

It turns out that there is an easy reduction from the problem SetCSP to that of ACAC, and therefore the proof that ACAC is MA-hard follows immediately.

Interestingly, the proof of containment in MA is rather simple for ACAC (it is an easy application of a known result from random walk theory.) Finally, using the same reduction, we arrive at a proof that the SetCSP problem is also in MA<sup>5</sup>.

We stress that we present all proofs here avoiding any quantum notation or quantum jargon whatsoever. The main contribution of this work is in the definitions themselves, initiated by the small but important conceptual idea of the translation mentioned above; this translation thus provides two new, combinatorial, and natural MA-complete problems, which we believe are amenable to research in a language familiar to (classical) computer scientists.

We now describe the problems. In the ACAC problem, we consider an exponentially large graph, accompanied with a function on the vertices that marks some of them. Both graph and the function on the vertices are given implicitly (and succinctly) by a polynomial size circuit. We then ask if there exists a connected component of the graph that is “clean” (meaning that all of its vertices are *unmarked*) or if the graph is  $\varepsilon$ -far from having this property. The notion of “far” is defined as follows: every set of vertices which is close to being a connected component (i.e. its expansion is smaller than  $\varepsilon$ ) must have at least an  $\varepsilon$ -fraction of its vertices marked. In other words, either a set is  $\varepsilon$ -far from a connected component (i.e. has large expansion) or at least  $\varepsilon$  fraction of its vertices are marked. We call this problem *approximately-clean approximate-connected-component* (ACAC $_{\varepsilon}$ ).

Our second MA-complete problem, called the *Set-Constraint Satisfaction Problem*, or SetCSP, is a somewhat unexpected generalization of the standard Constraint Satisfaction Problem (CSP). While a constraint in CSP acts on a single string (deciding if it is valid or not), the generalized constraints act on *sets* of strings. We call the generalized constraints *set-constraints* (see Section 4 for the exact definition of a set-constraint.). The input to the SetCSP problem is a collection of such set-constraints, and the output is whether there is a set of strings  $S$  that satisfies *all* set-constraints in this collection, or any set of strings  $S$  is  $\varepsilon$ -far from satisfying this set-constraint collection (see Definitions 10 and 16 for formal definitions of “satisfying a set-constraint” and “far”). We denote this problem by SetCSP $_{\varepsilon}$ .

Following the ideas of [8, 17], we show the following three claims: *i*) for every inverse polynomial  $\varepsilon$ , we have that ACAC $_{\varepsilon}$  is in MA (Corollary 8); *ii*) there exists an inverse polynomial function  $\varepsilon = \varepsilon(n)$  such that SetCSP $_{\varepsilon}$  is MA-hard (Lemma 17); and *iii*) for all functions  $\varepsilon = \varepsilon(n) > 0$ , there is a polynomial-time reduction from SetCSP $_{\varepsilon}$  to ACAC $_{\varepsilon/2}$ . Together, these facts imply the following results (which are proven in Section 5).

<sup>5</sup> This in fact gives a significantly simpler version than [7] of the proof of containment in MA, in the restricted case of uniform stoquastic Hamiltonians.

► **Theorem 1.** *There exists some inverse polynomial  $p(x) = \Theta(1/x^3)$  such that for every inverse polynomial  $p' < p$ , the problems  $\text{SetCSP}_{p'(m)}$  and  $\text{ACAC}_{p'(m)}$  are MA-complete, where  $m$  is the size of the  $\text{SetCSP}$  or  $\text{ACAC}$  instance.*

We stress that the definitions of both these problems and the proofs presented in this paper only use standard combinatorial concepts from set- and graph-theory.

## 1.1 Conclusions, future work and open questions

In a similar way to what is done in this note, we claim that it is possible to translate several other results from the stoquastic local Hamiltonian language to the  $\text{SetCSP}$  language, bringing us to very interesting conclusions.

First, one could strengthen the MA-hardness of  $\text{SetCSP}$  (Lemma 17), whose proof is a translation of the proof of the quantum Cook-Levin theorem [17], to the restricted case in which the constraints are set on a 2D lattice. To do this, one could consider the result of [3] where they prove the QMA-hardness of local Hamiltonians on a 2D lattice (considering only the restricted family of stoquastic Hamiltonians), and translate it to the  $\text{SetCSP}$  language. In this way, it can be proven that  $\text{SetCSP}_\varepsilon$  with inverse polynomial  $\varepsilon$ , and with set-constraints arranged on a 2D-lattice with constant size alphabet is still MA-hard.<sup>6</sup> We omit here the details of the proof, since it is a straight forward translation of [3], similarly to what's done in Lemma 17. This leads to the following statement.

► **Lemma 2.** *There exists some inverse polynomial  $p$  such that for every inverse polynomial  $p' < p$ ,  $\text{SetCSP}_{p'}$  is MA-hard even when each bit participates in  $O(1)$  set-constraints, and each set-constraint acts on  $O(1)$  bits.*

We also claim that the main result of [1], which states that some problem called stoquastic local Hamiltonian with constant gap is in NP, can be translated to  $\text{SetCSP}$  language, using again the same translation. This means that the *gapped* version of the  $\text{SetCSP}$  problem is in NP. By the gapped version of the problem, we mean  $\text{SetCSP}_\varepsilon$  when  $\varepsilon$  is a constant; and where we also require that the locality  $k$  (number of bits in a set-constraint) and degree  $d$  (number of set-constraints each bit participates in) are bounded from above by a constant. More concretely, we have the following.

► **Lemma 3.** *For any constant  $\varepsilon$  and constants  $k$  and  $d$ ,  $\text{SetCSP}_\varepsilon$  is in NP if each bit participates in  $d$  set-constraints, and each set-constraint acts on  $k$  bits.*

Together these two results lead to a very important and surprising equivalence<sup>7</sup>:

**Proposition.** MA=NP iff there is a gap amplification reduction<sup>8</sup> for  $\text{SetCSP}$ . The existence of a gap amplification reduction means that there exists a constant  $\varepsilon > 0$ , such that for every inverse polynomial  $p$ , there is a polynomial time reduction from  $\text{SetCSP}_{p(n)}$  to  $\text{SetCSP}_\varepsilon$ .

We note that it is easy to see that MA=NP implies such gap-amplification for  $\text{SetCSP}$ : if MA = NP, then we can reduce  $\text{SetCSP}_{1/poly}$ , which is in MA-complete by Theorem 1, to  $\text{CSP}_{O(1)}$ , which is NP-complete by the PCP theorem [10]; then, since every CSP instance

<sup>6</sup> We conjecture that this hardness result works even with binary alphabet and we leave such a statement for future work.

<sup>7</sup> This equivalence was highlighted in [1] in a quantum language of stoquastic Hamiltonians

<sup>8</sup> In the same sense as Dinur's gap amplification reduction for CSP[10]

is also a **SetCSP** instance with the same parameters, we have the gap-amplification. It is the other direction of deriving  $\text{MA}=\text{NP}$  from gap-amplification for **SetCSP**, that is the new contribution. This implication suggests a new path to the long standing open problem of derandomizing  $\text{MA}$ .

Finally, we leave as an open problem deriving such a natural statement regarding gap amplification for the **ACAC** problem. Though the two problems are technically very related, defining a natural restricted version of the gapped **ACAC** problem, so that the [1] result would apply to show containment in  $\text{NP}$  (similarly to **SetCSP** with constant  $\varepsilon$ , locality  $k$  and degree  $d$ ) remains open. The problem is that the locality notions don't have a very natural analogues in the graph language of **ACAC**.

**Organization.** We provide notation and a few basic notions in Section 2. In Section 3, we define the approximately-clean approximate-connected-component problem and prove its containment in  $\text{MA}$ . The definition of **SetCSP** and the proof of its  $\text{MA}$ -hardness are in Section 4. The reduction from **SetCSP** to **ACAC** appear in Section 5.

## 2 Preliminaries

For  $n \in \mathbb{N}^+$ , we denote  $[n] = \{0, \dots, n-1\}$ . For any  $n$ -bit string, we index its bits from 0 to  $n-1$ . For  $x \in \{0,1\}^n$  and  $J \subseteq [n]$ , we denote  $x|_J$  as the substring of  $x$  on the positions contained in  $J$ . For  $x \in \{0,1\}^{|J|}$ ,  $y = \{0,1\}^{n-|J|}$  and  $J \subseteq [n]$ , we define  $x^J y^{\bar{J}}$  to be the unique  $n$ -bit string  $w$  such that  $w|_J = x$  and  $w|_{\bar{J}} = y$ , where  $\bar{J} = [n] \setminus J$ . For two strings,  $x$  and  $y$ , we denote by  $x || y$  their concatenation, and  $|x|$  denotes the number of bits in  $x$ .

### 2.1 Complexity classes

A (promise) problem  $A = (A_{\text{yes}}, A_{\text{no}})$  consists of two non-intersecting sets  $A_{\text{yes}}, A_{\text{no}} \subseteq \{0,1\}^*$ .

For the definitions of the two main complexity classes that are considered in this work,  $\text{NP}$  and  $\text{MA}$ , we refer to the full version of this paper [2].

The standard definition of  $\text{MA}$  [4] requires yes-instances to be accepted with probability at least  $\frac{2}{3}$ , but it has been shown that there is no change in the computational power if we require the verification algorithm to always accept yes-instances [21, 12].

### 2.2 Reversible circuits

It is folklore that the verification algorithms for  $\text{NP}$  and  $\text{MA}$  can be converted into a uniform family of polynomial-size Boolean circuits, made of reversible gates,  $\{\text{NOT}, \text{CNOT}, \text{CCNOT}\}$  by making use of additional auxiliary bits initialized to 0, with only linear overhead. For randomized circuits, we can also assume the circuit uses only reversible gates, by assuming that the random bits are part of input. See the full version of this paper [2] for more details.

Let  $G \in \{\text{NOT}, \text{CNOT}, \text{CCNOT}\}$  be a gate to be applied on the set of bits  $J$  out of some  $n$ -bit input  $x$ . We slightly abuse notation (but make it much shorter!) and denote  $G(x) \stackrel{\text{def}}{=} x|_{\bar{J}} G(z|_J)^J$ . Namely, we understand the action of the  $k$ -bit gate  $G$  on an  $n > k$  bit string  $x$  by applying  $G$  only on the relevant bits, and leaving all other bits intact.

### 3 Approximately-clean approximate-connected-component problem

In this section, our goal is to present the approximately-clean approximate-connected-component problem and prove its containment in MA. But before that, we explain the *exact* version of this problem.

For a fixed parameter  $n$ , we consider a graph  $G$  of  $2^n$  nodes, which is described by a classical circuit  $C_G$  of size (number of gates)  $\text{poly}(n)$ , as follows. For simplicity, we represent each vertex of  $G$  as an  $n$ -bit string, and  $C_G$ , on input  $x \in \{0,1\}^n$ , outputs the (polynomially-many) neighbors of  $x$  in  $G$ .<sup>9</sup> We are also given a circuit  $C_M$ , which when given input  $x \in \{0,1\}^n$ , outputs a bit indicating whether the vertex  $x$  is marked or not.

We define the clean connected component problem (CCC) which, on input  $(C_G, C_M)$ , asks if  $G$  has a connected component where *all vertices are unmarked* or if all connected components have at least one marked element. We give now the formal definition of this problem.

► **Definition 4** (Clean connected component(CCC)). *Fix some parameter  $n$ . An instance of the clean connected component problem consists of two classical  $\text{poly}(n)$ -size circuits  $C_G$  and  $C_M$ .  $C_G$  consists of a circuit succinctly representing a graph with  $G = (V, E)$  where  $V = \{0,1\}^n$  and each vertex has degree at most  $\text{poly}(n)$ . On input  $x \in \{0,1\}^n$ ,  $C_G$  outputs all of the neighbors of  $x$ .  $C_M$  is a circuit that on input  $x \in \{0,1\}^n$ , outputs a bit. The problem then consists of distinguishing the two following cases:*

**Yes.** *There exists one non empty connected component of  $G$  such that  $C_M$  outputs 0 on all of its vertices.*

**No.** *In every connected component of  $G$ , there is at least one vertex for which  $C_M$  outputs 1.*

We show in the full version of this paper [2] that this problem is PSPACE-complete. Our focus here is to study the *approximate* version of CCC, where we ask whether  $G$  has a clean connected component or it is “far” from having this property, meaning that for every set of vertices  $S$ , the boundary<sup>10</sup> of  $S$  is at least  $\varepsilon|S|$  (and therefore  $S$  is far from being a connected component), or, if the boundary is small (i.e.  $S$  is close to a connected component) it contains at least  $\varepsilon|S|$  marked elements. Here is the definition of the problem.

► **Definition 5** (Approximately-clean approximate-connected-component(ACAC $_\varepsilon$ )). *Same as Definition 4 with the following difference for no-instances:*

**No.** *For all non empty  $S \subseteq V$ , we have that either*

$$|\partial_G(S)| \geq \varepsilon|S| \quad \text{or} \quad |\{x \in S : C_M(x) = 1\}| \geq \varepsilon|S|.$$

In the remainder of this section, we show that ACAC $_\varepsilon$  is in MA for every inverse polynomial  $\varepsilon$ .

#### 3.1 Inclusion in MA

The idea of the proof is as follows. The prover sends a vertex that belongs to the (supposed) clean connected component and then the verifier performs a random-walk on  $G$  for sufficiently (but still polynomially-many) steps and rejects if the random walk encounters a marked element.

<sup>9</sup> We notice that usually we succinctly describe graphs by considering a circuit that, on input  $(x, y)$ , outputs 1 iff  $x$  is connected to  $y$ . In our result it is crucial that given  $x$ , we are able to efficiently compute *all* of its neighbors.

<sup>10</sup> The boundary of a set of vertices  $S \subseteq V$  is defined as  $\partial_G(S) = \{\{u, v\} \in E : u \in S, v \notin S\}$ .



In order to prove that this verification algorithm is correct, we first prove a technical lemma regarding the random-walk on no-instances.

► **Lemma 6** (In NO-instances the random walk reaches marked nodes quickly). *Let  $\varepsilon$  be an inverse polynomial function of  $n$ . If  $(C_G, C_M)$  is a no-instance of  $\text{ACAC}_\varepsilon$ , then there exist polynomials  $q_1$  and  $q_2$  such that for every  $x \in \{0, 1\}^n$ , a  $q_1(n)$ -step lazy random walk<sup>11</sup> starting at  $x$  reaches a marked vertex with probability at least  $\frac{1}{q_2(n)}$ .*

**Proof.** Let  $x$  be some initial (unmarked) vertex (notice that we can assume that  $x$  is unmarked since otherwise the lazy random walk reaches a marked vertex with probability 1). Let  $G_x$  be the connected component of  $x$  in  $G$  and  $V_x$  be the set of vertices in the connected component of  $x$ . We partition  $V_x$  into  $A$ , the unmarked vertices in  $V_x$  and  $B = V_x \setminus A$ , the marked vertices in  $V_x$ .

We want to upper bound the size of the edge boundary of any set  $S \subseteq A$  which contains only unmarked strings. We claim that by the conditions of the lemma, it must be that for any  $S \subseteq A$ , we have the following bound:

$$|\partial_G(S)| \geq \varepsilon|S|, \quad (1)$$

otherwise  $S$  would contradict the fact that  $(C_G, C_M)$  is a no-instance, since it only contains unmarked vertices.

We can now ask how fast does a lazy random walk starting from  $x$ , reach an element in  $B$ . This is a well known question from random walk theory, and it can be stated as follows.

► **Lemma 7** (Escaping time of high conductance subset). *Let  $G = (V = A \cup B, E)$  be a simple (no multiple edges) undirected connected graph, such that for every  $v \in A$   $d_G(v) \leq d$ , and such that for some  $\delta < \frac{1}{2}$ , for all  $A' \subseteq A$ , we have that  $|\partial_G(A')| \geq \delta|A'|$ . Then a  $\left(\frac{16d^2}{\delta^2} \ln \frac{2d|V|}{\delta}\right)$ -step lazy random walk starting in any  $v \in A$  reaches some vertex  $u \in B$  with probability at least  $\frac{\delta}{4d}$ .*

We defer the proof of this lemma to the full version of this paper [2] and we apply it using  $G = G_x$ ,  $A$ ,  $B$ ,  $d$  is the (poly( $n$ )-large) maximum degree of  $G$ , and  $\delta \geq \varepsilon$ . It follows that a  $\left(\frac{16d^2}{\varepsilon^2} (n + \ln(2d/\varepsilon))\right)$ -step lazy random walk starting on any  $x \in G_x$  reaches a marked vertex with probability at least  $\frac{\varepsilon}{4d}$ . ◀

From the previous lemma, we can easily achieve the following.

► **Corollary 8.** *For any inverse polynomial  $\varepsilon$ ,  $\text{ACAC}_\varepsilon$  is in MA.*

**Proof.** The witness for the  $\text{ACAC}$  instance consists of some string  $x$ , which is supposed to be in a clean connected component of  $G$ . Define  $q_1$  and  $q_2$  as the same polynomials of Lemma 6, namely, let  $q_1 = \left(\frac{16d^2}{\varepsilon^2} (n + \ln(2d/\varepsilon))\right)$  and  $q_2 = \frac{4d}{\varepsilon}$ . The MA-verification algorithm consists of repeating  $nq_2(n)$  times the following process: start from  $x$ , perform a  $q_1(n)$ -step lazy random walk in  $G$ , reject if any of these walks encounters a marked vertex, otherwise accept.

If  $(C_G, C_M)$  is a yes-instance, then the prover can provide a vertex  $x$  which belongs to the clean connected component. Any walk starting from  $x$  remains in its connected component and thus it will never encounter a marked vertex and the verifier accepts with probability 1.

<sup>11</sup>In a lazy random walk, at every step we stay in the current vertex with probability  $\frac{1}{2}$ , and with probability  $\frac{1}{2}$  we choose a random neighbor of the current vertex uniformly at random and move to it.



If  $(C_G, C_M)$  is a no-instance, from Lemma 6, each one of the random walks finds a marked vertex with probability at least  $\frac{1}{q_2(n)}$ . Thus, if we perform  $nq_2(n)$  lazy random walks, the probability that at least one of them finds a marked vertex is exponentially close to 1. ◀

#### 4 Set-Constraint Satisfaction Problem

In this section we present the Set Constraint Satisfaction Problem (SetCSP), and then prove its MA-hardness.

##### 4.1 Definition of the SetCSP problem

We start by recalling the standard Constraint Satisfaction Problem (CSP). We choose to present CSP in a way which is more adapted to our generalization (but still equivalent to the standard definition). An instance of  $k$ -CSP is a sequence of constraints  $C_1, \dots, C_m$ . In this paper, we see each constraint  $C_i$  as a tuple  $(J(C_i), Y(C_i))$ , where  $J(C_i) \subseteq [n]$  is a subset of at most  $k$  distinct elements of  $[n]$  (these are referred to as “the bits on which the constraint acts”) and  $Y(C_i)$  is a subset of  $k$ -bit strings, namely  $Y(C_i) \subseteq \{0, 1\}^{|J(C_i)|}$  (these are called the “allowed strings”). We say that a string  $x \in \{0, 1\}^n$  satisfies  $C_i$  if  $x|_{J(C_i)} \in Y(C_i)$ . The familiar problem of  $k$ -CSP, in this notation, is to decide whether there exists an  $n$ -bit string  $x$  which satisfies  $C_i$ , for all  $1 \leq i \leq m$ .

In  $k$ -SetCSP, our generalization of  $k$ -CSP, the constraints  $C_i$  are replaced by what we call “set-constraints” which are satisfied by (or alternatively, that *allow*), *sets* of strings  $S \subseteq \{0, 1\}^n$ .

► **Definition 9** (Set constraint). *A  $k$ -local set-constraint  $C$  consists of a) a tuple of  $k$  distinct elements of  $[n]$ , denoted  $J(C)$  (we have  $|J(C)| = k$ , and we refer to  $J(C)$  as the bits which the set-constraint  $C$  acts on), and b) a collection of sets of strings,  $Y(C) = \{Y_1, \dots, Y_\ell\}$ , where  $Y_i \subseteq \{0, 1\}^k$  is a set of  $k$ -bit strings and  $Y_i \cap Y_j = \emptyset$  for all distinct  $1 \leq i, j \leq \ell$ .*

► **Definition 10** (A set of strings satisfying a constraint). *We say that a set of strings  $S \subseteq \{0, 1\}^n$  satisfies the  $k$ -local set-constraint  $C$  if first, the  $k$ -bit restriction of any string in  $S$  to  $J(C)$  is contained in one of the sets in  $Y(C)$ , i.e., for all  $x \in S$ ,  $x|_{J(C)} \in \bigcup_j Y_j$ . Secondly we require that if  $x \in S$  and  $x|_{J(C)} \in Y_j$ , then for every  $y$  such that  $y|_{J(C)} \in Y_j$  and  $y|_{\overline{J(C)}} = x|_{\overline{J(C)}}$ , then  $y \in S$ . In other words, for any string  $s \in S$ , one can replace its  $k$ -bit restriction to the bits  $J(C)$ , which is a string in some  $Y_j \in Y(C)$ , with a different  $k$ -bit string in  $Y_j$ , and the resulting string  $s'$  must also be in  $S$ .*

An instance of  $k$ -SetCSP consists of  $m$  such  $k$ -local set-constraints, and we ask if there is some non-empty  $S \subseteq \{0, 1\}^n$  that satisfies each of the set constraints, or if any set  $S$  of  $n$ -bit strings is *far* from satisfying the collection of set-constraints. How to define *far*? We quantify the distance from satisfaction using a generalization of the familiar notation of *unsat* from PCP theory [10]; we denote the generalized notion by  $\text{set-unsat}(\mathcal{C}, S)$ . Intuitively, this quantity captures how much we need to modify  $S$  in order to satisfy the collection of set-constraints. Making this definition more precise will require some work; However we believe it already makes some sense intuitively, so we present the definition of the SetCSP problem now, and then provide the exact definition of  $\text{set-unsat}(\mathcal{C}, S)$  in Section 4.2.

► **Definition 11** ( $k$ -local Set Constraint Satisfaction problem ( $k$ -SetCSP $_\epsilon$ )). *Fix the two constants  $d, k \in \mathbb{N}^+$ , as well as a monotone function  $\epsilon : \mathbb{N}^+ \rightarrow (0, 1)$  to be some parameters of the problem. An instance to the  $k$ -local Set Constraint Satisfaction problem is a sequence*

of  $m(n)$   $k$ -local set-constraints  $\mathcal{C} = (C_1, \dots, C_m)$  on  $\{0, 1\}^n$ , where  $m$  is some polynomial in  $n$ . Under the promise that one of the following two holds, decide whether:

**Yes.** There exists a non-empty  $S \subseteq \{0, 1\}^n$  that satisfies all set constraints in  $\mathcal{C}$ , i.e.,  $\text{set-unsat}(\mathcal{C}, S) = 0$

**No.** For all  $S \subseteq \{0, 1\}^n$ ,  $\text{set-unsat}(\mathcal{C}, S) \geq \varepsilon(n)$ .

## 4.2 Satisfiability, frustration and the definition of $\text{set-unsat}(\mathcal{C}, S)$

We present some concepts required for the formal definition of  $\text{set-unsat}(\mathcal{C}, S)$ .

► **Definition 12** (*C*-Neighboring strings). Let  $C$  be some set-constraint. Two distinct strings  $x$  and  $y$  are said to be *C*-neighbors if  $x|_{\overline{J(C)}} = y|_{\overline{J(C)}}$  and  $x|_{J(C)}, y|_{J(C)} \in Y_i$ , for some  $Y_i \in Y(C)$ . We call a string  $x$  a *C*-neighbor of  $S$  if there exists a string  $y \in S$  such that  $x$  is a *C*-neighbor of  $y$ .

We also define the *C*-longing strings in a set of strings  $S$ : these are the strings that are in  $S$  but are *C*-neighbors of some string that is not in  $S$ .

► **Definition 13** (*C*-Longing strings). Given some set  $S \subseteq \{0, 1\}^n$  and a set-constraint  $C$ ,  $x \in S$  is a *C*-longing<sup>12</sup> string with respect to  $S$  if  $x$  is a *C*-neighbor of some  $y \notin S$ .

A useful definition is that of *bad* strings for some set-constraint  $C$ , which in short are the strings that do not appear in any subset of  $Y(C)$ .

► **Definition 14** (*C*-Bad string). Given a set-constraint  $C$ , with  $Y(C) = \{Y_1, \dots, Y_\ell\}$ , a string  $x \in \{0, 1\}^n$  is *C*-bad if  $x|_{J(C)} \notin \bigcup_i Y_i$ . We abuse the notation and whenever  $x$  is *C*-bad, we say  $x \notin Y(C)$ .

The following complementary definition will be useful:

► **Definition 15** (Good string). We say that a string is *C*-good if it is not *C*-bad. We say that it is a “good string for the set-constraint collection  $\mathcal{C}$ ” if it is *C*-good for all set-constraints  $C \in \mathcal{C}$ . When the collection  $\mathcal{C}$  is clear from context (as it is throughout this note), we omit mentioning of the set-constraints collection  $\mathcal{C}$  and just say that the string is “good”.

Given Definitions 13 and 14 above, it is easy to see that a set of strings  $S \subseteq \{0, 1\}^n$  satisfies a set-constraint  $C$  (by Definition 10) iff  $S$  contains no *C*-bad strings and no *C*-longing strings.

We now provide a way to quantify how far  $S$  is from satisfying  $C$ .

► **Definition 16** (Satisfiability of set-constraints). Let  $S \subseteq \{0, 1\}^n$  and  $C$  be a  $k$ -local set-constraint. Let  $B_C$  be the set of *C*-bad strings in  $S$  and  $L_C$  be the set of *C*-longing strings in  $S$ . Note that  $L_C \cap B_C = \emptyset$ . The  $\text{set-unsat}$  value (which we sometimes refer to as the *frustration*) of a set-constraint  $C$  with respect to  $S$ , is defined by

$$\text{set-unsat}(C, S) = \frac{|B_C|}{|S|} + \frac{|L_C|}{|S|} \quad (2)$$

<sup>12</sup> The term “*C*-longing” reflects the sentiment that the string  $x$  “wants” to be together with  $y$  in  $S$ ; the set constraint makes sure that there is an energy penalty if this is not the case.

Given a collection of  $m$   $k$ -local set-constraints  $\mathcal{C} = (C_1, \dots, C_m)$ , its **set-unsat** value (or frustration) with respect to  $S$  is defined as average frustration of the different set-constraints:

$$\text{set-unsat}(\mathcal{C}, S) = \frac{1}{m} \sum_{i=1}^m \text{set-unsat}(C_i, S). \quad (3)$$

We also define the frustration of the set collection  $\mathcal{C}$ :

$$\text{set-unsat}(\mathcal{C}) = \min_{\substack{S \subseteq \{0,1\}^n \\ S \neq \emptyset}} \{\text{set-unsat}(\mathcal{C}, S)\}. \quad (4)$$

We say that  $\mathcal{C}$  is satisfiable if  $\text{set-unsat}(\mathcal{C}) = 0$  and for  $\varepsilon > 0$ , we say that  $\mathcal{C}$  is  $\varepsilon$ -frustrated if  $\text{set-unsat}(\mathcal{C}) \geq \varepsilon$ .

Notice that the normalization factors in Equation (2) guarantees that the **set-unsat** value lies between 0 and 1.<sup>13</sup>

### 4.3 Intuition and standard CSP as special case

We can present a standard  $k$ -CSP instance consisting of constraints  $C_1, C_2, \dots, C_m$  as an instance of  $k$ -SetCSP in the following way: For each constraint  $C_\ell, \ell \in \{1, \dots, m\}$  out of the  $m$  constraints in the  $k$ -CSP instance, we consider the following set-constraint  $C'_\ell$ : for every  $k$ -bit string  $s$  that satisfies  $C_\ell$ , add the subset  $Y_s = \{s\}$  to  $C'_\ell$ . We arrive at a collection  $\mathcal{C}$  of  $m$  set-constraints, where each set-constraint  $C'_\ell$  in  $\mathcal{C}$  consists of single-string sets  $Y_i$  corresponding to all strings which satisfy  $C_\ell$ .

We claim that the resulting SetCSP instance has  $\text{set-unsat}(\mathcal{C}) = 0$  if and only if the original CSP instance was satisfiable. First, if the original CSP instance is satisfiable, we claim that for any satisfying string  $s$  we can define the set  $S = \{s\}$  consisting of that single string, and  $S$  indeed satisfies the collection of set-constraints defined above. To see this, note that in our case, there is no notion of  $C$ -neighbors (See Definition 12), since all  $Y_i$ 's contain only a single string. Hence, there are no longing strings; By the definition of **set-unsat** (Equation (2)) in this case  $\text{set-unsat}(\mathcal{C}, S) = 0$  if all strings in  $S$  are good for all set-constraints  $C'$ , namely each of them satisfies all constraints  $C$  in the original  $k$ -CSP instance, which is indeed the case if we pick a satisfying assignment. For the other direction, assume  $\text{set-unsat}(\mathcal{C}, S) = 0$  for some set  $S$ . This in particular means that all strings in  $S$  are  $C$ -good for all set-constraints in  $\mathcal{C}$ . By definition of our  $\mathcal{C}$ , this means any string  $s \in S$  is a satisfying assignment.

We give now some intuition about the **set-unsat** quantity (Equation (2)). We first note that it generalizes the by-now-standard notion of (un)satisfiability in CSP, which for a given string, counts the number of unsatisfied constraints, divided by  $m$ . We note that in Equation (2), if  $S = \{s\}$ , then  $|B_C|$  is either 0 or 1, depending on whether  $s$  satisfies the constraint or not. As previously remarked, in CSP the notion of neighboring and longing strings does not exist, and so  $L_C$  will always be empty. Thus, in the case where  $S = \{s\}$  and all set-constraints containing single strings, Equation (3) is indeed the number of violated constraints by the string, divided by  $m$  – which is exactly the standard CSP **unsat** used in PCP contexts [10]. (In the case of  $S$  containing more than one string, but all  $Y_i$ s are still single-strings, Equation (3) will just be the (non-interesting) average of the **unsat** of all strings in  $S$ ).

<sup>13</sup>The lower bound is trivial since the value cannot be negative. For the upper-bound, notice that  $B_C, L_C \subseteq S$  and  $B_C \cap L_C = \emptyset$ . This is because bad strings have no neighbors whereas longing strings do. Hence  $\frac{|B_C|}{|S|} + \frac{|L_C|}{|S|} \leq \frac{|S|}{|S|} = 1$ .

The interesting case is the general **SetCSP** case, when the  $Y_i$ 's contain more than a single string, i.e., when the notions of neighbors and longing strings become meaningful. In this case, the left term in the RHS of Equation (2) quantifies how far the set  $S$  is from the situation in which it contains only “good” (not “bad”) strings (this can be viewed as the standard requirement) but it adds to it the right term, which quantifies how far  $S$  is from being closed to the action of adding neighbors with respect to the set-constraints<sup>14</sup>. Loosely put, the generalization from constraints to set-constraints imposes strong “dependencies” between different strings, and the number of longing strings,  $L_C$ , counts to what extent these dependencies are violated.

#### 4.4 MA-hardness of **SetCSP**<sub>1/poly( $n$ )</sub>

In this subsection, we show the MA-hardness of **SetCSP**<sub>1/poly</sub>.

► **Lemma 17.** *There exists some inverse polynomial  $p(x) = \Theta(1/x^3)$  such that for every inverse polynomial  $p' < p$ , the problem **SetCSP** <sub>$p'(m)$</sub>  is MA-hard.*

To prove this lemma, we show how to reduce any language  $L \in \text{MA}$  to a 6-**SetCSP** instance. Our approach here is to “mimic” the Quantum Cook-Levin theorem due to Kitaev [17], but given that we only need to deal with set of strings and not arbitrary quantum states, our proof can in fact be stated in set-constraints language.

##### 4.4.1 Intuition for how set-constraints can check histories

In the celebrated proof of the (classical) Cook-Levin theorem, an instance of an NP-language is mapped to an instance for 3-SAT problem. More precisely, the verifier  $V$  of the NP-problem, which runs on an  $n$ -bit input string  $x$  and a  $\text{poly}(n)$  bit witness  $y$ , is mapped to a 3-SAT formula. To do this, a different variable is assigned to the value of each location of the tape of the Turing machine of  $V$  at any time step; these variables are used to keep track of the state of the computation of  $V$  (namely what is written on the tape) at the different time steps (see Figure 1 for an example). The formula acts on strings of bits, which can be viewed as assignments to *all* these variables; such an assignment encodes the *entire history* of a single possible computation. The clauses of the Boolean formula check if the history given by the assignment, indeed corresponds to a correct propagation of an *accepted* computation. More precisely, the constraints check that *a*) the assignment to the  $n$  Boolean variables at the first time step, associated with the input to the NP-problem, is indeed correct (namely equal to the true input  $x$ ); *b*) the assignment of the variables corresponding to any two subsequent time steps is consistent with a correct evolution of the appropriate gate in the computation; and *c*) the output bit, namely the value of the output variable in the *last* time step, is indeed *accept*. There exists a valid history which ends with accept (i.e.,  $x$  is in the language accepted by the verifier), iff the resulting  $k$ -CSP is satisfiable; in which case a satisfying assignment encodes a *history* of a correct computation of the verifier.

Now, suppose we want to apply a similar construction for some MA verification. The random bits are also given to the verification circuit as an input, and one could hope that the reduction of the Cook-Levin theorem would still work. However, the problem is that there could be *some* choice of random coins that makes the verification algorithm accept

<sup>14</sup>Notice that we could have chosen to define “far” here, by counting the number of strings *outside* of  $S$  that are neighbors of  $S$ . But since the degree of each string in the graph is bounded, the exact choice of the definition does not really matter; we chose the one presented here since it seems most natural.

even for no-instances, because the soundness parameter is not 0. Hence, the CSP would be satisfiable in this case, even though the input is a NO instance. It is thus not sufficient to verify the existence of a *single* valid history. In order to distinguish between YES and NO instances of the problem, we have to check in the YES case that *all* (or at least many of the) initializations of the random bits lead to accept. The key difficulty is how to check that not only a single string satisfies the constraints, but many.

This is exactly the reason for introducing the *set-constraints*. These set-constraints are able to verify that the random bits are indeed *uniformly random*; then the standard Cook-Levin approach described above can verify that (given the right witness) most of them leads to acceptance.

In order to implement such an approach, we first explain how to modify the original Cook-Levin proof, of the NP completeness of satisfiability, so that the strings that we check are not entire histories of a verification process of some NP problem; rather, the strings represent *snapshots* of the tape (or evaluations of all bits involved in the circuit at a certain time) for *different time-steps* of the computation. A satisfying assignment is no longer a single string but a whole collection of strings, denoted  $S$ , which would be the *collection of all snapshots of a single valid computation, at different times*.<sup>15</sup>

In this case (note that we have still not included random bits in the discussion) we need to show how to create *set-constraints* that verify that the set  $S$  really does contain *all* snapshots, and it also needs to verify that  $S$  contains nothing else. More precisely, we need to verify that *a)* the strings in  $S$  are consistent with being snapshots of a valid evolution of the computation *b)* the input is correct in the string corresponding to the first snapshot, and the output is accept in the string corresponding to the last snapshot, and *c)* the set  $S$  indeed contains the whole *history* of the computation, i.e. all the snapshots in a correct evolution, and without any missing step.

It turns out that this can be done using set-constraints; We depict the differences between the “original” Cook-Levin proof and the “set-constraints” one in Figure 1.

In reality, we need to do this not just for a single evolution but for the evolution over all possible assignments to the random strings. Moreover, we need to enforce that the random bits are indeed random; this requires further set-constraints (technically, this is done below in Equation (5)).

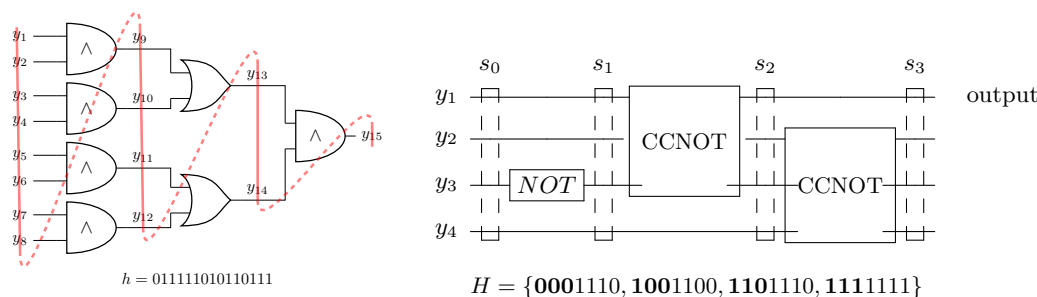
We note that the essence of the translation idea mentioned in the introduction appears already when there are no random bits involved at all; the reader is recommended to pretend that no random bits are used, at her first reading.

We next explain the reduction from MA-verification algorithms to 6-SetCSP in Section 4.4.2, and then prove its correctness in Section 4.4.3.

#### 4.4.2 The reduction

We assume, without loss of generality, that the MA verification algorithm is reversible, as described in Section 2.2: Given an input  $x \in \{0, 1\}^n$ , we assume a verification circuit  $C_x$  whose input is  $y \parallel 0^{a(n)} \parallel r$ , where  $y \in \{0, 1\}^{p(n)}$  is the polynomial-size MA witness, the middle register consists of the circuit auxiliary bits (needed for reversibility) and  $r \in \{0, 1\}^{q(n)}$  are the polynomially many random bits used during the MA verification. The circuit  $C_x$

<sup>15</sup> For quantum readers, we note that this reflects the main step in Kitaev’s modification of the Cook-Levin theorem, which enabled him to test that entangled quantum states evolved correctly.



■ **Figure 1** Comparison between the evolution of a circuit in the Cook-Levin proof and in our work.

consists of  $T$  gates  $G_1, \dots, G_T$ , where  $G_i \in \{\text{NOT}, \text{CNOT}, \text{CCNOT}\}$ .<sup>16</sup> At the end of the circuit, we assume WLOG that the first bit is the output bit. We describe now the reduction from the MA problem into a **SetCSP** instance  $\mathcal{C}$ . We will show in the next section that if there is an MA witness that makes the verification algorithm accept with probability 1, then  $\mathcal{C}$  is satisfiable, whereas if every witness makes the verifier reject with probability at least  $\frac{1}{2}$ , then  $\mathcal{C}$  is at least inverse polynomially frustrated.

We start with an MA verification circuit  $C_x$  (assumed to be reversible, as described in Section 2.2) acting on an input consisting of  $a(n)$  0-bits, witness  $y$  of  $p(n)$  bits and  $q(n)$  random bits. Thus, the number of bits which the reversible verification circuit acts on is  $w(n) = a(n) + p(n) + q(n)$ . The number of gates is  $T(n)$ ; denote these gates by  $G_1, \dots, G_T$ . Our set-constraint instance  $\mathcal{C}$  will act on strings of  $s(n) = T(n) + w(n)$  many bits. We omit  $n$  from such functions from now on, since it will be clear from the context.

We call the  $T$  first bits of such strings the *clock* register and the last  $w$  bits the *work* register. The work register comprises of three sub-registers: the witness register (first  $p(n)$  bits), the auxiliary register (middle  $a(n)$  bits) and the randomness register (last  $q(n)$  bits). We want to create set-constraints which force all the strings to be of the form  $z \in \{0, 1\}^s$  such that  $z_{[T]} = \text{unary}(t)$  for some  $t \in [T + 1]$  (where  $\text{unary}(t)$  denotes the integer  $t$  written in unary representation), and  $z_{[s] \setminus [T]}$  represents the *snapshot* of the computation at time  $t$  (as explained in Section 4.4.1 above) for the initial string  $y \parallel 0^{a(n)} \parallel r$ , for some witness  $y$  and some choice of random bits  $r$ .

<sup>16</sup> Recall that the gate CNOT on input  $a, b$  outputs  $a, b \oplus a$ , and the gate CCNOT on input bits  $a, b, c$ , outputs  $a, b, c \oplus ab$ .

### 36:14 Two Combinatorial MA-Complete Problems

We will construct  $\mathcal{C}$  in such a way that  $S \subseteq \{0, 1\}^s$  satisfies  $\mathcal{C}$  if and only if *i)* it contains *all* snapshots of the computation for some witness  $y$  and for *all* bit strings  $r$  input to the randomness register; and *ii)* the computations whose snapshots are contained in  $S$  are not only correct (meaning also that the auxiliary bits are all initialized to 0) but that they are *accepted* computations (meaning that the output is 1).

We do this by providing set-constraints of four types, as follows.

**Clock consistency.** We first impose that if  $z \in S$ , then  $z|_{[T]} = \text{unary}(t)$ , for some  $t \in [T + 1]$ .

Notice that a string is a valid unary encoding iff it does not contain 01 as a substring. To guarantee that the clock bits are consistent with some unary representation of an allowed  $t$ , we add for every  $t \in [T]$  the set-constraint  $C_t^{\text{clock}}$  defined by:

$$Y(C_t^{\text{clock}}) = (\{\{00\}, \{10\}, \{11\}\}) \text{ and } J(C_t^{\text{clock}}) = (t, t + 1).$$

Example: The string  $010^{T-2} || z$  is a bad string for  $C_1^{\text{clock}}$  since  $w|_{J(C_1^{\text{clock}})} = 01 \notin Y(C_1^{\text{clock}})$ .

**Initialization of Input bits and Random bits.** Here, we want to check that  $0^T || y || z || r$  is not in  $S$  whenever  $z \neq 0^a$ , which enforces the ancillary bits to be initialized to 0. In addition, we want to check that for any witness  $y$ , if one string of the form  $0^T || y || 0^a || r$  for some  $r$  is in  $S$ , then for all  $r' \in \{0, 1\}^q$ , we have  $0^T || y || 0^a || r'$  in  $S$ . In conclusion, we need to check two things: that all auxiliary bits are initialized to 0 and that all possible initializations of the random bits are present.

For each *auxiliary bit*  $j \in [a]$  we add a set-constraint  $C_j^{\text{aux}}$  and for every random bit  $j \in [q]$  we add the set constraint  $C_j^{\text{rand}}$  as follows.

We define

$$Y(C_j^{\text{aux}}) = (\{\{00\}, \{10\}, \{11\}\}) \text{ and } J(C_j^{\text{aux}}) = (0, T + p + j),$$

which forces that for  $t = 0$  (notice that the unique value of  $t$  for which  $\text{unary}(t)$  has the first bit 0 is  $t = 0$ ), the  $j$ -th auxiliary bit must be 0, because the string (01) is forbidden. For  $t \neq 0$  this is not enforced by allowing any value of the  $j$ -th auxiliary bit when the first clock bit is 1 (and therefore  $t \neq 0$ ).

Example: The string  $0^T || y || 10^{a-1} || r \in S$  is bad for  $C_0^{\text{aux}}$ .

For the *random bits*, we want to make sure that the  $j$ -th random bit has both values 0 and 1, over all the random bits. Therefore we define the constraints  $C_j^{\text{rand}}$  by

$$Y(C_j^{\text{rand}}) = (\{\{00, 01\}, \{10\}, \{11\}\}) \text{ and } J(C_j^{\text{rand}}) = (0, T + p + a + j). \quad (5)$$

These constraints will be useful in the following way. First, the propagation set-constraints that we will define soon, constrain  $S$  to make sure that there *exists* a string  $0^T || y || 0^a || r$  representing a snapshot at time 0 with some value of the random bits,  $r$ , which is indeed in  $S$ . The  $C_j^{\text{rand}}$  constraints enforce that given the existence of such a string in  $S$ , then for any other assignment to the random bits,  $r'$ , the string  $0^T || y || 0^a || r' \in S$ . Then, if many of these other strings are not in  $S$ , the frustration will be high.

Example: If  $s_1 = 0^T || y || 0^a || 0^r \in S$  but  $s_2 = 0^T || y || 0^a || 10^{r-1} \notin S$ , then  $s_1$  is a  $C_1^{\text{rand}}$ -longing string in  $S$  since  $s_1$  and  $s_2$  are  $C_1^{\text{rand}}$ -neighbors.



**Propagation.** Here we want to check that if  $\text{unary}(t-1) \parallel z \in S$  for some  $0 < t < T$ , then  $\text{unary}(t) \parallel G_t(z) \in S$ .

Let us consider the propagation constraint associated with the  $t$ -th timestamp (the one corresponding to the application of the  $t$ th gate,  $G_t$ ), for  $1 < t < T$ . Let us assume that the  $t$ -th gate acts on bits  $b_{t,1}, \dots, b_{t,k}$ . For this we add the set-constraint  $C_t^{\text{prop}}$  defined as follows:

$$Y(C_t^{\text{prop}}) = \bigcup_{z \in \{0,1\}^k} \{\{100 \parallel z, 110 \parallel G_t(z)\}\} \text{ and } J(C_t^{\text{prop}}) = (t-1, t, t+1, b_{t,1}, b_{t,2}, \dots, b_{t,k}).$$

For  $t = 1$  we simply erase the left clock bit from the above specification:

$$Y(C_1^{\text{prop}}) = \bigcup_{z \in \{0,1\}^k} \{\{00 \parallel z, 10 \parallel G_1(z)\}\} \text{ and } J(C_1^{\text{prop}}) = (1, 2, b_{1,1}, b_{1,2}, \dots, b_{1,k}).$$

Likewise if  $t = T$  erase the right most clock bit from the above.

Example: If  $s_1 = \text{unary}(t) \parallel z \in S$  but  $s_2 = \text{unary}(t+1) \parallel G_{t+1}(z) \notin S$ ,  $s_1$  is a  $C_{t+1}^{\text{prop}}$ -longing string in  $S$ , since the two strings are  $C_{t+1}^{\text{prop}}$ -neighbors.

**Output.** Finally, we need to check that for all strings of the form  $1^T \parallel z$ , the first bit of  $z$  is 1, namely, the last snapshot corresponds to accept. We define  $C^{\text{out}}$  such that

$$Y(C^{\text{out}}) = \{\{00\}, \{01\}, \{11\}\} \text{ and } J(C^{\text{out}}) = (T, T+1).$$

Here we use the fact that the  $T$ -th bit of  $\text{unary}(t)$  is 1 iff  $t = T$ . In this case, if this value is 1, we require that the output bit is 1, otherwise it could have any value.

Example: The string  $1^T \parallel 0 \parallel z$  is bad for  $C^{\text{out}}$ .

► **Remark 18.** We notice that for every set-constraint  $C$  that we constructed above, we have that  $Y(C)$  only contains sets of size 1 or 2. This property adds a bit more structure to SetCSP instances that are MA-hard, which could be useful in future work.

#### 4.4.3 Correctness

Given some MA-verification circuit  $C_x$ , we consider the following 6-SetCSP instance

$$\mathcal{C}_x = (C_1^{\text{clock}}, \dots, C_T^{\text{clock}}, C_1^{\text{aux}}, \dots, C_a^{\text{aux}}, C_1^{\text{rand}}, \dots, C_q^{\text{rand}}, C_1^{\text{prop}}, \dots, C_T^{\text{prop}}, C^{\text{out}}).$$

Let  $m$  be the number of terms in  $\mathcal{C}_x$  and when it is more convenient to us, we will refer to the set-constraints in  $\mathcal{C}_x$  as  $C_i$  for  $i \in [m]$ , where the terms have an arbitrary order. We show now that  $\mathcal{C}_x$  is satisfiable if  $x$  is a positive instance, and if  $x$  is a negative instance, then  $\mathcal{C}$  is at least  $\frac{1}{10(T+1)qm}$ -frustrated (where we remember that  $q$  is the number of random coins used by  $C_x$ ). Notice that  $m$ , the number of constraints in  $\mathcal{C}_x$ , is polynomial in  $T$ , which is also polynomial in  $|x|$ .

We start by proving completeness.

► **Lemma 19** (Yes-instances lead to satisfiable SetCSP instances). *If  $x \in \mathbf{L}$ , then  $\mathcal{C}_x$  is satisfiable.*

**Proof.** Let  $y$  be the witness that makes  $C_x$  accept with probability 1, and let

$$S = \{\text{unary}(t) \parallel G_t \dots G_1(y, 0^a, r) : r \in \{0,1\}^q, t \in [T+1]\}.$$

By construction, the initialization, clock and propagation constraints are satisfied by  $S$ . By the assumption that the MA verification circuit accepts with probability 1, the output constraints are also satisfied by  $S$ . ◀

Next we prove soundness.

► **Lemma 20** (NO-instances lead to frustrated SetCSP instances). *If  $x \notin L$ , then  $\text{set-unsat}(\mathcal{C}_x, S) \geq \frac{1}{10(T+1)qm}$  for every non-empty  $S \subseteq \{0, 1\}^n$ .*

**Proof.** Let  $S \subseteq \{0, 1\}^n$  be a non-empty set,  $B$  the set of bad strings in  $S$  (namely a string in  $S$  which is  $C$ -bad for at least one set-constraint  $C$ ) and  $L$  the set of longing strings in  $S$  (namely the strings in  $S$  which are  $C$ -longing for at least one set-constraint  $C$ ). Our goal here is to consider a partition  $\{K_i\}$  of  $S$ , such that for every  $K_i$ ,  $|K_i \cap (B \cup L)| \geq \frac{|K_i|}{10(T+1)q}$ , and from this we will show that  $\text{set-unsat}(\mathcal{C}_x, S) \geq \frac{1}{10(T+1)qm}$ .

Let us start by defining  $S_{ic} \subseteq S$  to be the subset of  $S$  with invalid clock register. Notice that every  $x \in S_{ic}$  is bad for at least one clock constraint and therefore  $|S_{ic} \cap B| = |S_{ic}|$ .

Now we notice that all other strings correspond to some valid clock register whose value (when read as a unary representation of some integer) is in  $[T + 1]$ . Let us partition the strings in  $S \setminus S_{ic}$  into disjoint sets  $H_1, \dots, H_\ell$  (which indicate different history-sets to which the strings belong) as follows. We define the *initial configuration* of some string in  $S \setminus S_{ic}$  like this. The string must be of the form  $\text{unary}(t) \parallel z$  for some  $t$  and  $z$ . Then  $\text{initial}(\text{unary}(t) \parallel z) = \text{unary}(0) \parallel G_1^{-1} \dots G_t^{-1}(z)$ , which is the assignment of the initial bits that leads to the configuration  $z$  at the  $t$ 'th step. We say then that two strings  $s_1, s_2 \in H_i$  iff  $\text{initial}(s_1) = \text{initial}(s_2)$  and we abuse the notation and call  $\text{initial}(H_i) = \text{initial}(s_1)$ . Notice that for  $i \neq j$ ,  $H_i$  and  $H_j$  are disjoint because the computation is reversible; thus the  $H_i$ 's constitute a partition of  $S \setminus S_{ic}$ . Notice also that each  $H_i$  contains at most  $T + 1$  strings, and the different strings in each  $H_i$  have different values of the clock register. We call these  $H_i$  *history-sets* for the reason that they correspond to a correct propagation of the computation of the circuit  $C_x$  for some initialization of all its bits.<sup>17</sup>

Let us first consider the history-sets whose initial configuration is not valid, i.e., it contains invalid (or non-zero) auxiliary bits:  $\mathbf{H}^{ia} = \{H_i : \text{initial}(H_i) = 0^T \parallel y \parallel z \parallel r \text{ for some } z \neq 0^a\}$ . We note that for any  $H_i \in \mathbf{H}^{ia}$ ,  $\text{initial}(H_i)$  is a  $C_j^{aux}$ -bad string in  $H_i$  for some  $j$ . If this string is in  $H_i$ , then we indeed have  $|H_i \cap (B \cup L)| \geq |H_i \cap B| \geq 1 \geq \frac{|H_i|}{T+1}$ . However, if  $H_i$  does not contain its initial string, then consider the minimal  $t$  such that  $\text{unary}(t) \parallel z \in H_i$  for some  $z$ , and by assumption we have  $t > 0$ . This means that the string  $\text{unary}(t) \parallel z$  is a  $C_t^{prop}$ -longing string, because it is a neighbor of  $\text{unary}(t-1) \parallel G_t^{-1}(z) \notin S$ . Hence, for such  $H_i$  we have  $|H_i \cap (B \cup L)| \geq |H_i \cap L| \geq 1 \geq \frac{|H_i|}{T+1}$ . This completes handling all the strings in  $S$  within a history set  $H_i$  with invalid auxiliary bits in  $\text{initial}(H_i)$ .

We now need to consider history sets in  $\{H_i\} \setminus \mathbf{H}^{ia}$ , namely, the history sets whose initial string is of the form  $0^T \parallel y \parallel 0^a \parallel r$  for some value of  $y$  and  $r$ . Let us group these history sets according to  $y$ , the value of the witness register. In other words, let us consider the sets of history sets:  $\mathbf{H}_y = \{H_i : H_i\text{'s initial string is of the form } 0^T \parallel y \parallel 0^a \parallel r\}$ . We fix now some  $y$  and the following arguments hold for each  $y$  separately. Notice that each  $H_i \in \mathbf{H}_y$  corresponds to a computation corresponding to a different initial random string for the witness  $y$ . Let us denote the union of strings in all sets in  $\mathbf{H}_y$  by  $\mathbf{S}_y = \bigcup_{H_i \in \mathbf{H}_y} \{s \mid s \in H_i\}$ . To finish handling all strings, we need to provide a bound on  $|\mathbf{S}_y \cap (L \cup B)|$  for all witnesses  $y$ .

To proceed, we need some additional notation. We note that  $\mathbf{H}_y$  can be written as the union of the history sets which contain their initial string, denoted  $\mathbf{H}_y^{\text{start}}$ , and the rest, denoted  $\mathbf{H}_y^{\text{nostart}}$ . We proceed by considering two cases separately:

<sup>17</sup> More precisely,  $H_i$  is a subset of the correct propagation because it involves only the snapshots in  $S$ .

1. Let us first consider the simpler case in which  $|\mathbf{H}_y^{nostart}| > \frac{|\mathbf{H}_y|}{10}$ . As above, we have that each  $H_i \in \mathbf{H}_y^{nostart}$  has a long string, and therefore we have that  $|\mathbf{S}_y \cap (L \cup B)| \geq |\mathbf{S}_y \cap L| \geq |\mathbf{H}_y^{nostart}| > \frac{|\mathbf{H}_y|}{10} \geq \frac{|\mathbf{S}_y|}{10(T+1)}$ . The last inequality is due to the fact that each  $H_i \in \mathbf{H}_y$  contains at most  $T+1$  strings. This finishes the treatment of all strings in  $\mathbf{S}$ , in the case  $|\mathbf{H}_y^{nostart}| > \frac{|\mathbf{H}_y|}{10}$ .
2. In the second case  $|\mathbf{H}_y^{start}| \geq \frac{9|\mathbf{H}_y|}{10}$ . We further denote  $\mathbf{S}_y^{init} = \{s | s = \text{initial}(H_i), H_i \in \mathbf{H}_y^{start}\}$  as the set of the initial strings of each  $H_i \in \mathbf{H}_y^{start}$ . Again (and for the last time) there are two cases.

- a. First, let us consider the case when  $|\mathbf{S}_y^{init}| \geq 2^{q-1}$ . This means that for this fixed  $y$ , for most values  $r$  of the random bits, the initial string  $0^T || y || 0^a || r$  of the history set corresponding to this  $y$  and  $r$  is present in  $\mathbf{S}_y$ . We use the facts that  $x \notin \mathbf{L}$ , and that at least  $2/3$  of the history sets must lead to rejection. From these observations, we will conclude that there will be either many bad strings due to the final accept constraint  $C^{out}$ , or many long strings.

Let  $\mathbf{Acc}_y = \{H_i \in \mathbf{H}_y : 1^T || z \in H_i \text{ and } z = 1 || z'\}$  be the set of history sets in  $\mathbf{H}_y$  that accept in the last step. We have that  $|\mathbf{Acc}_y|$  is at most the number of  $r \in \{0, 1\}^q$  which leads the circuit  $C_x$  to accept the witness  $y$ ; since  $x \notin \mathbf{L}$ , we have that the probability to accept for any  $y$  is most  $1/3$ . Hence  $|\mathbf{Acc}_y| \leq \frac{2^q}{3} \leq 2 \frac{|\mathbf{H}_y|}{3} \leq \frac{20|\mathbf{H}_y^{start}|}{27}$ , where we used the fact that  $2^{q-1} \leq |\mathbf{S}_y^{init}| = |\mathbf{H}_y^{start}| \leq |\mathbf{H}_y|$ , and in the last inequality, the fact that we are in the case  $|\mathbf{H}_y^{start}| \geq 9|\mathbf{H}_y|/10$ . Hence, there are at least  $\frac{2^q}{10}$  history sets in  $\mathbf{H}_y^{start}$  which do not end in accept. Such  $H_i$  either contains the string  $1^T || 0 || z$  (which is bad for  $C^{out}$ ), or does not contain a final state at all, namely does not contain a state of the form  $1^T || z$  for some  $z$ , resulting in a long string. Thus, there are at least  $\frac{2^q}{10}$  strings in  $|\mathbf{S}_y \cap (B \cup L)|$ ; and since  $|\mathbf{S}_y| \leq (T+1)|\mathbf{H}_y|$  and  $|\mathbf{H}_y| \leq 2^q$ , resulting in  $|\mathbf{S}_y| \leq 2^q(T+1)$ , we have that  $|\mathbf{S}_y \cap (B \cup L)| \geq \frac{2^q}{10} \geq \frac{|\mathbf{S}_y|}{10(T+1)}$ .

- b. Finally, let us consider the case where  $|\mathbf{S}_y^{init}| \leq 2^{q-1}$ . This is where we will need to apply conductance arguments. Let  $G_y^0$  be the subgraph of  $G_C^{18}$  induced by the vertices  $R_y = \{0^T || y || 0^a || r : r \in \{0, 1\}^q\}$ . Notice that  $G_y^0$  is isomorphic to the  $q$ -dimensional hypercube. This is true because the only remaining edges on  $G_y^0$  come from the set-constraints  $C_j^{rand}$ . Notice also that  $\mathbf{S}_y^{init}$  is a subset of the vertices of  $G_y^0$ . We now use the fact that the conductance of the  $q$ -dimensional hypercube is  $\frac{1}{q}$ . Applying this lemma to the graph  $G_y^0$  and the subset of its vertices,  $\mathbf{S}_y^{init}$  we conclude that  $\frac{|\partial_{G_y^0}(\mathbf{S}_y^{init})|}{q|\mathbf{S}_y^{init}|} \geq \frac{1}{q}$ , where we have used the fact that all vertices in  $G_y^0$  have the same degree,  $q$ , and the fact that we are now considering the case  $|\mathbf{S}_y^{init}| \leq 2^{q-1}$ . We can conclude then that there exists at least  $|\mathbf{S}_y^{init}|$  edges in the cut  $(\mathbf{S}_y^{init}, R_y \setminus \mathbf{S}_y^{init})$ . Since each vertex in  $G_y^0$  (in particular, each vertex in  $\mathbf{S}_y^{init}$ ) has  $q$  neighbors, it means that there exists at least  $\frac{|\mathbf{S}_y^{init}|}{q}$  long strings in  $\mathbf{S}_y^{init}$ . We conclude this case by noticing that

$$|\mathbf{S}_y \cap (L \cup B)| \geq |\mathbf{S}_y \cap L| \geq \frac{|\mathbf{S}_y^{init}|}{q} = \frac{|\mathbf{H}_y^{start}|}{q} \geq \frac{9|\mathbf{H}_y|}{10q} \geq \frac{9|\mathbf{S}_y|}{10q(T+1)},$$

where in the second inequality we use the fact that  $\mathbf{S}_y^{init} \subseteq \mathbf{S}_y$  and there are at least  $\frac{|\mathbf{S}_y^{init}|}{q}$  long strings in  $\mathbf{S}_y^{init}$ , the equality follows since  $|\mathbf{H}_y^{start}| = |\mathbf{S}_y^{init}|$ , the third inequality follows from our assumption that  $|\mathbf{H}_y^{start}| \geq \frac{9|\mathbf{H}_y|}{10}$  and finally we have that  $|H_i| \leq T+1$  (and therefore  $|\mathbf{H}_y| \geq \frac{|\mathbf{S}_y|}{T+1}$ ).

<sup>18</sup> Here, we use the same notation as Section 3.1.

## 36:18 Two Combinatorial MA-Complete Problems

To finish the proof, notice that  $S = S_{ic} \cup \bigcup_{H \in \mathbf{H}^{ia}} H \cup \bigcup_y \mathbf{S}_y$ . Since each of these subsets has at least a  $\frac{1}{10(T+1)q}$ -fraction of bad strings or longing strings, we have that  $|B \cup L| \geq \frac{|S|}{10(T+1)q}$ . It follows that

$$\text{set-unsat}(\mathcal{C}_x, S) = \frac{1}{m} \sum_i \left( \frac{|B_{C_i}|}{|S|} + \frac{|L_{C_i}|}{|S|} \right) \geq \frac{|B|}{m|S|} + \frac{|L|}{m|S|} \geq \frac{|B \cup L|}{m|S|} \geq \frac{1}{10(T+1)qm},$$

finishing the proof.  $\blacktriangleleft$

From the two previous lemmas, we have the following.

► **Lemma 17 (restated).** *There exists some inverse polynomial  $p(x) = \Theta(1/x^3)$  such that for every inverse polynomial  $p' < p$ , the problem  $\text{SetCSP}_{p'(m)}$  is MA-hard.*

**Proof.** It follows directly from Lemma 19, together with the fact that, regarding the parameters in Lemma 20, we have that  $T, q \leq m$ .  $\blacktriangleleft$

### 5 Reduction from SetCSP to ACAC

In this section we reduce the SetCSP problem to the ACAC, showing the containment of SetCSP in MA and the MA-hardness of ACAC.

Before showing the reduction, we prove a technical lemma that shows how, for a fixed  $S$ , the value of  $\text{set-unsat}(\mathcal{C}, S)$  and the number of bad and longing strings for  $\mathcal{C}$  are related.

► **Lemma 21.** *For some fixed non-empty  $S \subset \{0, 1\}^n$ , let  $B_{\mathcal{C}} = \bigcup_i B_{C_i}$  and  $L_{\mathcal{C}} = \bigcup_i L_{C_i}$ , the union of bad and longing strings for all set-constraints in  $\mathcal{C}$ , respectively. We have that  $\text{set-unsat}(\mathcal{C}, S) \leq \frac{1}{|S|}(|B_{\mathcal{C}}| + |L_{\mathcal{C}}|)$ .*

**Proof.** By definition of  $\text{set-unsat}(\mathcal{C}, S)$  and  $\text{set-unsat}(C_i, S)$ , we have that

$$\begin{aligned} \text{set-unsat}(\mathcal{C}, S) &= \frac{1}{m} \sum_{i=1}^m \text{set-unsat}(C_i, S) = \frac{1}{m|S|} \sum_{i=1}^m |B_{C_i}| + |L_{C_i}| \leq \frac{1}{m|S|} \sum_{i=1}^m |B_{\mathcal{C}}| + |L_{\mathcal{C}}| \\ &= \frac{1}{|S|}(|B_{\mathcal{C}}| + |L_{\mathcal{C}}|), \end{aligned}$$

where in the inequality we use the fact that  $B_{C_i} \subseteq B_{\mathcal{C}}$  and  $L_{C_i} \subseteq L_{\mathcal{C}}$ .  $\blacktriangleleft$

For some inverse polynomial  $\varepsilon$ , we consider an instance  $\mathcal{C}$  of  $k\text{-SetCSP}_{\varepsilon}$ . From  $\mathcal{C}$ , we construct the graph  $G_{\mathcal{C}} = (\{0, 1\}^n, E)$ , where  $(x, y) \in E$  if there exists a set-constraint  $C \in \mathcal{C}$  such that  $x$  and  $y$  are  $C$ -neighbors. We can define  $C_{G_{\mathcal{C}}}$  that on input  $x \in \{0, 1\}^n$ , outputs all neighbors of  $x$  by inspecting all set-constraints of  $\mathcal{C}$ . Finally, we define  $C_M$  as the circuit that on input  $x \in \{0, 1\}^n$ , outputs if  $x$  is a bad string for  $\mathcal{C}$ , again by inspecting all of its set-constraints.

► **Lemma 22 (Reduction from SetCSP to ACAC).** *For every  $\varepsilon$  we have that:*

- *If  $\mathcal{C} = (C_1, \dots, C_m)$  is a yes-instance of  $k\text{-SetCSP}_{\varepsilon}$ , then  $(C_{G_{\mathcal{C}}}, C_M)$  is a yes-instance of  $\text{ACAC}_{\varepsilon/2}$ .*
- *If  $\mathcal{C} = (C_1, \dots, C_m)$  is a no-instance of  $k\text{-SetCSP}_{\varepsilon}$ , then  $(C_{G_{\mathcal{C}}}, C_M)$  is a no-instance of  $\text{ACAC}_{\varepsilon/2}$ .*

**Proof.** To prove the first part of our statement, we show that a non-empty  $S \subseteq \{0, 1\}^n$  such that  $\text{set-unsat}(S, \mathcal{C}) = 0$  implies that the connected component of any string  $x \in S$  in  $G_{\mathcal{C}}$  contains only good strings. To show this, we notice that  $S$  is a union of connected components of  $G_{\mathcal{C}}$ . In this case, any of these connected components imply that  $(C_{G_{\mathcal{C}}}, C_M)$  is a yes-instance of ACAC.

Suppose towards contradiction that there exists a string  $x$  in  $S$  which is connected to a string  $y$  outside of  $S$  via an edge in  $G_C$ ; that means that  $x$  and  $y$  are  $C$ -neighbors for some set-constraint  $C$ . But this means that  $x$  is a  $C$ -longing string for  $S$ , and this contradicts  $\text{set-unsat}(S, C) = 0$ . We finish this part of the proof by stressing that, by assumption, no elements in  $S$  are marked, otherwise there would be a bad string in it.

For the second part, we show that if there is a set of vertices  $S$  such that  $\partial(S, \bar{S}) < \varepsilon|S|/2$  on  $G_C$  and the number of marked elements in  $S$  is strictly less than  $\varepsilon|S|/2$ , then  $\text{set-unsat}(S, C) < \varepsilon$ . In this case, if  $C$  is a no-instance of  $k\text{-SetCSP}_\varepsilon$ , then  $(C_{G_C}, C_M)$  must be a no-instance of  $\text{ACAC}_{\varepsilon/2}$ .

Notice that if there are at most  $\varepsilon|S|/2$  edges between  $S$  and  $\bar{S}$ , then there are at most  $\varepsilon|S|/2$  vertices in  $S$  that are connected to  $\bar{S}$  and, by definition of the edges in  $G_C$ , we have that  $S$  has at most  $\varepsilon|S|/2$   $C$ -longing strings. We also have that the number of bad strings in  $S$  is, by definition, the number of marked elements which is also strictly less than  $\varepsilon|S|/2$ . Therefore, by Lemma 21, we have that  $\text{set-unsat}(C, S) < \varepsilon$ . ◀

We can now finally prove Theorem 1:

▶ **Theorem 1 (restated).** *There exists some inverse polynomial  $p(x) = \Theta(1/x^3)$  such that for every inverse polynomial  $p' < p$ , the problems  $\text{SetCSP}_{p'(m)}$  and  $\text{ACAC}_{p'(m)}$  are MA-complete, where  $m$  is the size of the  $\text{SetCSP}$  or  $\text{ACAC}$  instance.*

**Proof.** From Lemma 17 we have that for some  $\tilde{p} = \Theta(1/x^3)$ ,  $\text{SetCSP}_{\tilde{p}}$  is MA-hard<sup>19</sup> and from Corollary 8 we have that  $\text{ACAC}_{\tilde{p}/2}$  is in MA.

In Lemma 22, we show a reduction  $\text{SetCSP}_{\tilde{p}}$  to  $\text{ACAC}_{\tilde{p}/2}$ , which implies, together with the aforementioned results, that  $\text{SetCSP}_{\tilde{p}}$  is in MA and that  $\text{ACAC}_{\tilde{p}/2}$  is MA-hard. Therefore, we can pick  $p(x) = \tilde{p}(x)/2$  and our statement holds. ◀

## References

- 1 Dorit Aharonov and Alex B. Grilo. Stoquastic PCP vs. Randomness. In *FOCS 2019*, 2019. [arXiv:1901.05270](#).
- 2 Dorit Aharonov and Alex B. Grilo. Two combinatorial MA-complete problems. *arXiv preprint*, 2020. [arXiv:2003.13065](#).
- 3 Dorit Aharonov, Wim van Dam, Julia Kempe, Zeph Landau, Seth Lloyd, and Oded Regev. Adiabatic quantum computation is equivalent to standard quantum computation. *SIAM Review*, 50(4):755–787, 2008.
- 4 László Babai. Trading group theory for randomness. In *Proceedings of the 17th Annual ACM Symposium on Theory of Computing*, pages 421–429, 1985.
- 5 László Babai, Lance Fortnow, Noam Nisan, and Avi Wigderson. Bpp has subexponential time simulations unless exptime has publishable proofs. *Comput. Complex.*, 3(4):307–318, October 1993. [doi:10.1007/BF01275486](#).
- 6 Sergey Bravyi. Monte Carlo simulation of stoquastic Hamiltonians. *arXiv preprint*, 2014. [arXiv:1402.2295](#).
- 7 Sergey Bravyi, Arvid J. Bessen, and Barbara M. Terhal. Merlin-arthur games and stoquastic complexity. *arXiv preprint*, 2006. [arXiv:quant-ph/0611021](#).
- 8 Sergey Bravyi and Barbara M. Terhal. Complexity of stoquastic frustration-free hamiltonians. *SIAM J. Comput.*, 39(4):1462–1485, 2009.

<sup>19</sup>Notice that we slightly abuse the notation here: in Lemma 17, we define the hardness in respect to the parameter  $m$ , the number of clauses; here, we call  $m$  the size of the  $\text{SetCSP}$  instance, which is lower-bounded by its number of clauses.

- 9 Elizabeth Crosson, Dave Bacon, and Kenneth R. Brown. Making classical ground-state spin computing fault-tolerant. *Phys. Rev. E*, 82:031106, September 2010.
- 10 Irit Dinur. The PCP Theorem by Gap Amplification. *Journal of the ACM*, 54(3), 2007.
- 11 Andrew Drucker. A pcg characterization of am. In *Proceedings of the 38th International Colloquium Conference on Automata, Languages and Programming - Volume Part I*, ICALP'11, 2011.
- 12 Oded Goldreich and David Zuckerman. Another proof that  $BPP \subseteq PH$  (and more). In *Studies in Complexity and Cryptography*, pages 40–53. Springer-Verlag, 2011.
- 13 Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4), March 1999.
- 14 Russell Impagliazzo, Valentine Kabanets, and Avi Wigderson. In search of an easy witness: exponential time vs. probabilistic polynomial time. *J. Comput. Syst. Sci.*, 65(4):672–694, 2002.
- 15 Russell Impagliazzo and Avi Wigderson.  $P = BPP$  if  $E$  requires exponential circuits: Derandomizing the xor lemma. In *Proceedings of the Twenty-ninth Annual ACM Symposium on Theory of Computing*, STOC '97, pages 220–229. ACM, 1997. doi:10.1145/258533.258590.
- 16 Valentine Kabanets and Russell Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Computational Complexity*, 13(1):1–46, December 2004.
- 17 Alexei Kitaev, A Shen, and M N Vyalyi. *Classical and quantum computation*. Graduate studies in mathematics. American mathematical society, Providence (R.I.), 2002. URL: <http://opac.inria.fr/record=b1100148>.
- 18 Noam Nisan and Avi Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994.
- 19 Peter Shor and Ryan Williams. Mathoverflow: Complete problems for randomized complexity classes. URL: <https://mathoverflow.net/questions/34469/complete-problems-for-randomized-complexity-classes>. Accessed: 2019-01-14.
- 20 Madhu Sudan, Luca Trevisan, and Salil Vadhan. Pseudorandom generators without the xor lemma. *Journal of Computer and System Sciences*, 62(2):236–266, 2001.
- 21 Stathis Zachos and Martin Furer. Probabilistic quantifiers vs. distrustful adversaries. In *Seventh Conference on Foundations of Software Technology and Theoretical Computer Science*, pages 443–455, 1987.

# Delegated Stochastic Probing

Curtis Bechtel 

Department of Computer Science, University of Southern California, Los Angeles, USA  
bechtel@usc.edu

Shaddin Dughmi 

Department of Computer Science, University of Southern California, Los Angeles, USA  
shaddin@usc.edu

---

## Abstract

Delegation covers a broad class of problems in which a principal doesn't have the resources or expertise necessary to complete a task by themselves, so they delegate the task to an agent whose interests may not be aligned with their own. Stochastic probing describes problems in which we are tasked with maximizing expected utility by "probing" known distributions for acceptable solutions subject to certain constraints. In this work, we combine the concepts of delegation and stochastic probing into a single mechanism design framework which we term delegated stochastic probing. We study how much a principal loses by delegating a stochastic probing problem, compared to their utility in the non-delegated solution. Our model and results are heavily inspired by the work of Kleinberg and Kleinberg in "Delegated Search Approximates Efficient Search." Building on their work, we show that there exists a connection between delegated stochastic probing and generalized prophet inequalities, which provides us with constant-factor deterministic mechanisms for a large class of delegated stochastic probing problems. We also explore randomized mechanisms in a simple delegated probing setting, and show that they outperform deterministic mechanisms in some instances but not in the worst case.

**2012 ACM Subject Classification** Theory of computation → Algorithmic mechanism design

**Keywords and phrases** Delegation, Mechanism Design, Algorithmic Game Theory

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.37

**Funding** *Curtis Bechtel*: Supported by NSF Grants CCF-1350900 and CCF-2009060.

*Shaddin Dughmi*: Supported by NSF CAREER Award CCF-1350900 and NSF Grant CCF-2009060.

## 1 Introduction

The division of labor and responsibility, based on expertise, is a defining characteristic of efficient organizations and productive economies. In the context of economic decision-making, such division often manifests through *delegation* scenarios of the following form: A decision maker (the *principal*), facing a multivariate decision beset by constraints and uncertainties, tasks an expert (the *agent*) with collecting data, exploring the space of feasible decisions, and proposing a solution.

As a running example, consider the leadership of a firm delegating some or all of its hiring decisions to an outside recruitment agency. When the principal and the agent have misaligned utilities – such as when the agency must balance the firm's preferences with its own preferences over, or obligations towards, potential hires – the principal faces a mechanism design problem termed *optimal delegation* (see e.g. [14, 3]). When the underlying optimization problem involves multiple inter-dependent decisions, such as when hiring a team which must collectively cover a particular set of skills, and when data collection is constrained by logistical or budget considerations, the problem being delegated fits in the framework of *stochastic probing*, broadly construed (see e.g. [26]).



© Curtis Bechtel and Shaddin Dughmi;

licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 37; pp. 37:1–37:19

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



The present paper is concerned with the above-described marriage of optimal delegation and stochastic probing. We restrict attention to protocols without payments, drawing our inspiration from the recent work of Kleinberg and Kleinberg [16]. The underlying (non-delegated) problem faced by the principal in their “distributional model” is the following: facing  $n$  i.i.d rewards, select the ex-post best draw. As for their “binary model”, there are  $n$  random rewards with binary support, and a cost associated with sampling each; the goal is to adaptively sample the rewards and select one, with the goal of maximizing the ex-post selected reward less sampling costs. For both models, they show that delegating the problem results in a loss of at most half the principal’s utility. Their analysis in both cases is through a reduction to the (classical) single-choice *prophet inequality* problem, and in particular to the threshold stopping rule of Samuel-Cahn [25].

Both the distributional and binary models of [16] can be viewed as stochastic probing problems, the former being trivial in the absence of delegation, and the latter corresponding to a special case of the well-studied box problem of Weitzman [27]. A number of stochastic probing problems have been known to reduce to *contention resolution schemes* (e.g. [10, 11, 7, 1, 9]), which in turn reduce to generalizations of the prophet inequality [21]. This suggests that the results of [16] might apply more broadly.

It is this suggestive thread which we pull on in this paper, unraveling what is indeed a broader phenomenon. We study optimal delegation for a fairly general class of stochastic probing problems with combinatorial constraints, and obtain delegation mechanisms which approximate, up to a constant, the principal’s non-delegated utility. Building on recent progress in the literature on stochastic optimization, our results reduce delegated stochastic probing to *generalized prophet inequalities* of a particular “greedy” form, as well as to the notion of *adaptivity gap* (e.g. [4, 5]).

## 1.1 Our Model

Our model features a collection of *elements*, each of which is associated with a (random) *utility* for each of the principal and the agent. We assume that different elements are independently distributed, though the principal’s and the agent’s utilities for the same element may be correlated. We allow constraining both the sampled and the selected set of elements via *outer* and *inner* constraints, respectively. Each constraint is a downwards-closed set system on the ground set of elements. A *probing algorithm* for an instance of our model adaptively *probes* some set of elements subject to the outer constraint, learning their associated utilities in the process. The algorithm then selects as its *solution* a subset of the probed elements satisfying the inner constraint. We assume that, for both the principal and the agent, utility for a solution is the sum of its per-element utilities.

To situate the non-game-theoretic component of our model within the literature on stochastic probing problems, note that we allow an arbitrary utility distribution for each element, rather than a binary-support distribution characterizing “feasibility”. Moreover, unlike “probe and commit” models, we also allow our algorithm to select its solution after all probes are complete. In both these respects, our model is akin to the stochastic multi-value probing model of [5]. As for our game-theoretic modeling, we assume that the utility distributions, as well as the inner and outer constraints, are common knowledge. The realized utilities, however, are only observed by the agent upon probing.

In the traditional (non-delegation) setting, the principal implements the probing algorithm optimizing her own utility, in expectation. In the delegation setting, the principal and agent engage in the following Stackelberg game. The principal moves first by *committing* to a *policy*, or *mechanism*. Such a policy is a (possibly randomized) map from a set of *signals*

to solutions satisfying the inner constraint, with each element in the solution labeled with its (presumptive) utility for both the principal and the agent. Moving second, the agent probes some set of elements subject to the outer constraint, and maps the observed utilities to a signal. The outcome of the game is then the solution which results from applying the principal's policy to the agent's signal. We assume that the principal and agent utilities are additive across elements in the solution, so long as it is labeled with the true per-element utilities. Otherwise, we assume that the principal can detect this discrepancy and effectively “quit” the game, imposing a utility of zero for both parties. We adopt the perspective of the principal, who seeks a policy maximizing her expected utility. The agent, naturally, responds with a strategy maximizing his own expected utility given the policy.

By an argument analogous to that in [16], which we prove in our general setting for completeness' sake, we can restrict attention to *single-proposal mechanisms*. In a deterministic single-proposal mechanism, the set of signals is a “menu” of *acceptable* (labeled) solutions satisfying the inner constraint, as well as a “null” signal which in our setting we can take to be the empty set. The agent, facing such a mechanism, without loss simply implements a probing algorithm to compute a “proposed” solution, tagging each element in the solution with its observed utilities, and ensuring that the solution is acceptable to the principal. We also consider randomized single-proposal mechanisms, where the menu consists of acceptable *lotteries* (i.e., distributions) over (labeled) solutions, and an agent's probing algorithm proposes a lottery on the menu.

## 1.2 Our Results

We study delegation mechanisms which approximate the principal's non-delegated utility. We refer to the best multiplicative approximation factor as the *delegation gap* of the associated instance.

Our main set of results concern the design of deterministic single-proposal mechanisms which prove constant delegation gaps for natural classes of inner and outer constraints. Our approach is modular, and reduces a (constructive)  $\alpha\beta$  bound on the delegation gap to a (constructive)  $\alpha$  generalized prophet inequality of a particular form on the inner constraint, and a (constructive) bound of  $\beta$  on the adaptivity gap associated with the outer constraint and the rank function of the inner constraint. Drawing on recent work in [9], which derives prophet inequalities of our required form, and in [4, 5], which bounds the adaptivity gap, we obtain constant bounds on the delegation gap for instances of our model with a variety of inner and outer constraints such as matroids and their intersections, among others.

We also begin an exploration of randomized single-proposal mechanisms, where the principal's menu consists of acceptable lotteries over solutions. We show that, even in the simple setting of no outer constraint and a 1-uniform inner constraint, there are instances for which randomized mechanisms significantly outperform optimal deterministic ones. Nevertheless, there exist worst-case instances where both deterministic and randomized mechanisms suffer a  $1/2$  delegation gap. We leave open whether randomized mechanisms can lead to better bounds on the worst-case delegation gap for more intricate classes of inner and outer constraints.

## 1.3 Additional Discussion of Related Work

Since the economic literature on delegation is extensive, we only describe a select sample here. The groundwork for the formal study of optimal delegation in economics was initially laid by Holstrom [14, 13]. Subsequent work in economics has considered a variety of optimization

problems as the task being delegated (e.g. [2, 23, 3]). We mention the work of Kovac and Mylovanov [18] as being related to our results in Section 5: To our knowledge, they were the first to examine the power of randomized mechanisms for delegation.

Most relevant to the present paper is the aforementioned work of Kleinberg and Kleinberg [16], who examine approximations for optimal delegation. Their distributional model is closely related to the model of Armstrong and Vickers [3], and the optimization problem being delegated in their binary model is a special case of Weitzman’s box problem [27]. Both optimization problems fit nicely in the general literature on stochastic probing (see e.g. [26]), motivating our examination of delegated stochastic probing more broadly.

Also related is the recent work of Khodabakhsh et al [15], who consider a very general model of delegation with discrete actions and states of the world, and an agent who fully observes the state (no outer constraints or sampling costs). They show optimal delegation to be NP-hard and examine limited “bias” assumptions under which simple threshold mechanisms are approximately optimal. Notably, they don’t impose sampling constraints on the agent and their approximations are with respect to the optimal delegation policy rather than the optimal non-delegated policy. For these reasons, our results are not directly comparable.

The optimization problems being delegated in our model fit in the broad class of *stochastic probing* problems. We do not attempt a thorough accounting of this literature, and instead refer the reader to related work discussions in [26, 5]. To our knowledge, the term “stochastic probing” was originally coined by Gupta and Nagarajan [10], though their binary probe-and-commit model is quite different from ours. More closely related to us are the models of [5, 4], which capture stochastic probing problems with multi-valued reward distributions, no commitment, and combinatorial inner and outer constraints.

As previously mentioned, our work draws on the literature on prophet inequalities. The foundational result in this setting is the (single-choice) prophet inequality of Krengel, Sucheston, and Garling [19, 20]. *Generalized prophet inequalities*, with respect to various combinatorial constraints, adversaries, and arrival models, have received much attention in the last decade (e.g. [12, 17, 8, 9]); the associated body of work is large, and we recommend the survey by [22]. Closely related to generalized prophet inequalities are *contention resolution schemes* (see e.g. [6, 9, 1]), with reductions going in both directions [21]. Key to our results are the “greedy” generalized prophet inequalities, derived through “greedy” contention resolution, by Feldman et al [9].

Finally, we briefly elaborate on the relationship between our model and the two models of Kleinberg and Kleinberg [16]. The natural variant of their binary model which replaces sampling costs with combinatorial constraints on the set of samples (outer constraints, in our nomenclature) fits squarely in our model. Their distributional model, which allows  $n$  i.i.d. samples from a distribution over utility pairs, initially appears to be a special case of ours. However, our principal is afforded additional power through their ability to distinguish elements by name alone. Nevertheless, we recover their main result as a special case of ours by observing that our mechanism treats elements symmetrically.

## 2 Preliminaries

Sections 2.1, 2.2, and 2.3 include brief introductions to some of the key ideas and notations used in this paper. Notably, Section 2.2 defines the key notion of “greedy” prophet inequalities.

## 2.1 Set Systems

A *set system* is a pair  $(E, \mathcal{I})$  where  $E$  is a finite set of *elements* and  $\mathcal{I} \subseteq 2^E$  is a family of *feasible* sets. We focus on *downwards-closed* set systems, satisfying the following two conditions: (1)  $\emptyset \in \mathcal{I}$ , i.e. the empty set is feasible, and (2) if  $T \in \mathcal{I}$  then  $S \in \mathcal{I}$  for all  $S \subseteq T$ , i.e. any subset of a feasible set is feasible. Matroids, matching constraints, and knapsack constraints are all examples of downwards-closed set systems.

For a set system  $\mathcal{M} = (E, \mathcal{I})$  and  $F \subseteq E$ , we use  $\mathcal{M}|F = (F, \mathcal{I} \cap 2^F)$  to denote the *restriction* of  $\mathcal{M}$  to  $F$ .

## 2.2 Prophet Inequalities

A *generalized prophet inequality problem* is given by a set system  $\mathcal{M} = (E, \mathcal{I})$ , and for each element  $e \in E$  an independent random variable  $X_e$  supported on the nonnegative real numbers. Here we adopt the perspective of a *gambler*, who is given  $\mathcal{M}$  and the distributions of the random variables  $\{X_e\}_{e \in E}$  in advance, then encounters the elements  $E$  in an order chosen by an *adversary*. On encountering  $e$ , the gambler observes the realization  $x_e$  of the random variable  $X_e$ , and must immediately decide whether to *accept*  $e$ , subject to the accepted set  $S$  of elements remaining feasible in  $\mathcal{M}$ . The gambler seeks to maximize his utility  $x(S) = \sum_{e \in S} x_e$ , and in particular to compete with a *prophet* who knows the realization of all random variables in advance. If the gambler can guarantee an  $\alpha$  fraction of the prophet's utility in expectation, we say that we obtain a generalized prophet inequality with a factor of  $\alpha$ .

For each possible realization  $x_e$  of  $X_e$ , we refer to the pair  $(e, x_e) \in E \times \mathbb{R}_+$  as an *outcome*. When the gambler accepts  $e \in E$  given a realization  $x_e$  of  $X_e$ , we also say the gambler accepts the outcome  $(e, x_e)$ .

Although it is most common to consider an adversary who fixes an order of the elements upfront, some recent work has investigated much more powerful adversaries [17, 9]. In this paper, we are interested in the *almighty adversary*, who knows in advance the realizations of all random variables as well as any random coin flips used by the gambler's strategy. The almighty adversary can perfectly predict the future and choose a truly worst-case ordering.

Key to our results is the notion of a “greedy” strategy for the gambler. We take inspiration from [9], who defined greedy online contention resolution schemes, and extend their definition to prophet inequality problems.

► **Definition 2.1.** Fix any instance of a generalized prophet inequality problem. A greedy strategy for the gambler is described by a downwards-closed family  $\mathcal{A} \subseteq 2^{E \times \mathbb{R}_+}$  of sets of outcomes. A gambler following greedy strategy  $\mathcal{A}$  accepts an outcome  $(e, x_e)$  if and only if the set of all accepted outcomes remains in  $\mathcal{A}$ .

We note that Samuel-Cahn's [25] threshold rule for the single-choice prophet inequality is greedy, and its competitive factor of  $\frac{1}{2}$  holds for the almighty adversary [24]. More generally, Feldman et al. [9] show that there exist constant-factor greedy prophet inequalities against the almighty adversary for many classes of constraints.

## 2.3 Adaptivity Gap

Another key notion we will use is the adaptivity gap for stochastic set function optimization problems. For a detailed introduction, see [4].

We consider maximizing a stochastic set function  $f : 2^E \rightarrow \mathbb{R}_+$  constrained by a downwards-closed set system  $\mathcal{M} = (E, \mathcal{I})$ . We assume  $f$  is determined by a collection  $\{X_e\}_{e \in E}$  of independent random variables, with the stipulation that  $f(S)$  does not depend

on any random variables  $X_e$  for which  $e \notin S$ .<sup>1</sup> We are tasked with “probing” some  $S \subseteq E$ , feasible for  $\mathcal{M}$ , with the goal of maximizing  $f(S)$ . An *adaptive* algorithm for this problem probes elements one at a time, where probing  $e$  results in learning the realization of  $X_e$ . Such an algorithm can use the realizations of probed variables to decide on a next element to probe. A *non-adaptive* algorithm chooses the set  $S$  all at once, independently of the random variables  $\{X_e\}_{e \in E}$ . The *adaptivity gap* is the minimum (worst-case) ratio of the expected value of the optimal non-adaptive algorithm versus the expected value of the optimal adaptive algorithm.

In [4], Asadpour and Nazerzadeh showed that the adaptivity gap for instances with monotone submodular functions and matroid constraints is  $1 - \frac{1}{e}$ . Furthermore, they provided an efficient non-adaptive algorithm that achieves this bound. Finally, in [5], Bradac et al. showed that the adaptivity gap is constant for instances with “prefix-closed” constraints (which include all downward-closed constraints) and functions that are the weighted rank function of the intersection of a constant number of matroids.

### 3 Model

#### 3.1 Formal Definition

► **Definition 3.1.** An instance  $I$  of the delegated stochastic probing problem consists of: two players, which we will call the principal and the agent; a ground set of elements  $E$ ; mutually independent distributions  $\mu_e$  with support in  $\mathbb{R}_+ \times \mathbb{R}_+$  for each element  $e \in E$ ; an outer constraint  $\mathcal{M}_{out} = (E, \mathcal{I}_{out})$  with feasible sets  $\mathcal{I}_{out}$ ; and an inner constraint  $\mathcal{M}_{in} = (E, \mathcal{I}_{in})$  with feasible sets  $\mathcal{I}_{in}$ .

Given such an instance, we will additionally define:  $(X_e, Y_e) \sim \mu_e$  as random variables denoting the utilities for the principal and agent of element  $e$ ;  $\Omega$  as the set of *outcomes*  $(e, x, y)$  for all  $e \in E$  and all  $(x, y) \in \text{supp}(\mu_e)$ ; and  $\Omega_{in} \subseteq 2^\Omega$  as the family of all sets of outcomes whose elements are distinct and feasible in the inner constraint. For convenience, we will also overload notation by considering  $x$  and  $y$  to be utility functions for the principal and agent. Given any subset of outcomes  $T \subseteq \Omega$ , let  $x(T) = \sum_{(e,x,y) \in T} x$  and  $y(T) = \sum_{(e,x,y) \in T} y$  be the total utility of outcomes in  $T$ . Similarly for any subset of elements  $F \subseteq E$ , let  $x(F) = \sum_{e \in F} X_e$  and  $y(F) = \sum_{e \in F} Y_e$  be random variables representing the randomized total utility of elements in  $F$ .

A natural mechanism that the principal might choose to implement is called a single-proposal mechanism. Here, the principal describes the space of solutions she is willing to accept, and then the agent uses this information to search the solution space and propose a single feasible solution.

In the deterministic single-proposal setting, the principal first commits to a family of sets of outcomes  $\mathcal{R} \subseteq \Omega_{in}$  and announces  $\mathcal{R}$  to the agent. The sets in  $\mathcal{R}$  are called *acceptable*, and the principal’s choice of  $\mathcal{R}$  is called their *policy* (or *mechanism*). After learning  $\mathcal{R}$ , the agent will select elements to probe, so long as each element is probed at most once and the set of probed elements is feasible in  $\mathcal{M}_{out}$ . We allow the agent to probe adaptively, deciding what to do next based on previously probed elements. Let’s say that they probe elements  $F \subseteq E$  and obtain outcomes  $S \subseteq \Omega$ . The agent will then choose some set of outcomes  $T \subseteq \Omega$  and *propose* it to the principal. If  $T$  is acceptable and also a subset of  $S$  then the principal and agent receive  $x(T)$  and  $y(T)$  utility, respectively. Otherwise, they both receive 0 utility.

<sup>1</sup> In other words, one can evaluate  $f(S)$  given access to the realizations of the random variables  $\{X_e\}_{e \in S}$ .

In the above-described mechanism design setting, we assume that both the principal and agent act to maximize their expected utility. We also assume that all parameters of the problem, except for the realizations of the random variables, are common knowledge.

We note that, similar to the setup in [16], our model assumes that our agent cannot benefit from lying, say by labeling an element  $e$  with utilities other than  $X_e$  and  $Y_e$ , or by proposing an element he has not probed. We argue that this is a natural assumption to make: In many applications we foresee (e.g., a firm hiring an employee, or exploring some mergers), a proposal will be accompanied by an easy to verify proof of the claimed utilities (e.g., in the form of a CV for the applicant, or a detailed analysis of the merger).

As in [16], we compare delegation mechanisms against the optimal *non-delegated* strategy. By non-delegated strategy, we mean the strategy of the principal when they act as both the principal and agent (i.e. they have power to probe and propose as well as accept outcomes).

Given any  $F \subseteq E$ , let  $u(F)$  be the optimal utility of the non-delegating principal when they probe elements in  $F$  and accept their own favorite set of outcomes, and let  $v_{\mathcal{R}}(F)$  be the utility of the delegating principal with policy  $\mathcal{R}$  when the agent probes elements in  $F$  and proposes their favorite acceptable set of outcomes. We can write  $u$  and  $v_{\mathcal{R}}$  as

$$u(F) = \max_{G \subseteq F, G \in \mathcal{I}_{\text{in}}} x(G)$$

$$v_{\mathcal{R}}(F) = x \left( \operatorname{argmax}_{G \subseteq F, \Omega_G \in \mathcal{R}} y(G) \right),$$

where  $\Omega_G \subseteq \Omega$  is the set of outcomes from the probed set of elements  $G$ . In the case of ties in the definition of  $v_{\mathcal{R}}$ , our results hold for arbitrary (even adversarial) tie-breaking.

► **Definition 3.2.** Fix any instance of delegated stochastic probing. Let  $F^*$  be a random variable containing the elements probed by an optimal adaptive non-delegating principal, and let  $F_{\mathcal{R}}^*$  be a random variable containing the elements probed by an optimal adaptive agent under policy  $\mathcal{R}$ . Then for any policy  $\mathcal{R}$  and  $\alpha \in [0, 1]$ , we say that  $\mathcal{R}$  is an  $\alpha$ -policy for this instance if

$$\mathbb{E} v_{\mathcal{R}}(F_{\mathcal{R}}^*) \geq \alpha \mathbb{E} u(F^*).$$

► **Definition 3.3.** The delegation gap of a family of instances of delegated stochastic probing is the minimum, over all instances in the family, of the maximum  $\alpha$  such that there exists an  $\alpha$ -policy for that instance. This gap measures the fraction of the principal's non-delegated utility they can achieve when delegating.

## 3.2 Signaling Mechanisms

Having formally defined the model, we will now describe a broad generalization of single-proposal mechanisms, called *signaling mechanisms*, and show that these mechanisms don't provide the principal with any additional power. Note that this discussion is inspired by Section 2.2 from [16], and we simply extend their work to our model.

A signaling mechanism allows the principal to ask the agent for more (or different) information than just a proposed solution. The principal will then take this information and transform it into a solution, which they will accept. One might suspect that expanding the space of mechanisms in this way would give the principal more power. However, as we will show, this isn't the case even for a broad class of delegation models, which we will now define formally.

► **Definition 3.4.** *An instance of the generalized delegation problem consists of two players called the principal and the agent, a state space  $S$ , a solution space  $\Psi$ , a set  $\mathcal{P}$  of probing strategies for the agent, a signaling function  $\sigma$  which maps  $\mathcal{P} \times S$  to strings, a utility function  $x : S \times \mathcal{P} \times \Psi \rightarrow \mathbb{R}_+$  for the principal, and a utility function  $y : S \times \mathcal{P} \times \Psi \rightarrow \mathbb{R}_+$  for the agent. We require that there is a null solution  $\perp \in \Psi$  such that  $x_{s,p}(\perp) = y_{s,p}(\perp) = 0$  for all  $s \in S$  and  $p \in \mathcal{P}$ .*

We assume the state of the world is some  $s \in S$  a-priori unknown to the principal and the agent, though they may have prior information. The agent obtains information about  $s$  by applying a probing strategy  $p \in \mathcal{P}$  to obtain a signal  $\sigma_p(s)$ . For a state  $s \in S$ , a probing strategy  $p \in \mathcal{P}$  chosen by the agent, and a solution  $\psi \in \Psi$ , we associate a utility of  $x_{s,p}(\psi)$  and  $y_{s,p}(\psi)$  for the principal and the agent, respectively.

We note that the above definition generalizes the delegation problems of Definition 3.1. In particular: the state space  $S$  represents all possible realizations of per-element utilities of the principal and the agent; the solution space  $\Psi$  is the family of feasible subsets of outcomes  $\Omega_{\text{in}}$ , where  $\perp$  is the empty set of outcomes;  $\mathcal{P}$  corresponds to probing algorithms which respect the outer constraint;  $\sigma_p(s)$  is the set of outcomes obtained by invoking algorithm  $p$  in state  $s$ ; both utility functions depend on the state  $s \in S$  and the probing algorithm  $p \in \mathcal{P}$ , evaluating to 0 for solutions  $\psi$  that are inconsistent with the state  $s$ , or if the probing algorithm  $p$  applied to  $s$  does not probe the elements in  $\psi$ .

Given a generalized delegation problem, we define signaling mechanisms as follows.

► **Definition 3.5.** *Fix some instance of the generalized delegation problem. A signaling mechanism proceeds in the following manner. The principal starts by choosing some signal space  $\Sigma$  of strings and a solution function  $\psi : \Sigma \rightarrow \Psi$ , and the agent responds by choosing a probing strategy  $p \in \mathcal{P}$  and a reporting function  $\tau$  from strings to  $\Sigma$ . Once these choices have been made, the agent will probe the underlying state  $s$  to obtain a signal  $\sigma = \sigma_p(s)$ , then transform this into a new signal  $\tau = \tau(\sigma)$  which he reports to the principal. The principal maps the reported signal to a solution  $\psi(\tau)$ , which they will accept.*

Notice that this model can be made to capture the design of randomized delegation mechanisms by extending  $\Psi$  to the space  $\Delta(\Psi)$  of distributions (henceforth *lotteries*) over solutions, and extending both utility functions to lotteries by taking expectations.

We contrast this broad definition of signaling mechanisms with the comparatively simple single-proposal mechanisms.

► **Definition 3.6.** *Fix an instance of the generalized delegation problem. A single-proposal mechanism is a special case of signaling mechanism in which the principal chooses some set  $\mathcal{R} \subseteq \Psi$  of acceptable outcomes, then sets  $\Sigma = \Psi$  and  $\psi(R) = R$  if  $R \in \mathcal{R}$  and  $\psi(R) = \perp$  otherwise.*

Intuitively, in a single proposal mechanism the principal declares a menu of acceptable solutions. The agent then proposes a solution, which is accepted if it is on the menu, and replaced with the null solution otherwise. Now we will show that single-proposal mechanisms are just as powerful as signaling mechanisms. In particular, for every signaling mechanism there is a single-proposal mechanism which selects the same solution and the same probing strategy for each state of nature, at equilibrium. This lemma is a simple extension of [16, Lemma 1] to the our generalized delegation model.

► **Lemma 3.7.** *Fix an instance of the generalized delegation problem, as well as the agent's prior distribution  $\mu$  on states  $S$ . For any signaling mechanism  $M = (\Sigma, \psi)$  and a corresponding best response strategy  $(p, \tau)$  for the agent, there exists a single-proposal mechanism  $M' = (\Sigma', \psi')$  and a corresponding best response  $(p, \tau')$  such that  $(\psi \circ \tau \circ \sigma_p)(s) = (\psi' \circ \tau' \circ \sigma_p)(s)$  for all states  $s \in S$ .*



**Proof.** Take any signaling mechanism  $M = (\Sigma, \psi)$  with best response  $(p, \tau)$  by the agent. Let  $\mathcal{R} = \psi(\Sigma)$  be the set of all possible outputs from this mechanism and let  $M' = (\Sigma', \psi')$  be the single-proposal mechanism defined by  $\mathcal{R}$ , i.e.  $\Sigma' = \Psi$  and  $\psi'$  is such that  $\psi'(R) = R$  if  $R \in \mathcal{R}$  and  $\psi'(R) = \perp$  otherwise. Finally, let  $\tau' = \psi \circ \tau$ .

Notice that the range of  $\tau'$  is contained in  $\psi(\Sigma) = \mathcal{R}$ , so by definition of  $\psi'$  and  $\tau'$  it follows that  $\psi \circ \tau = \psi' \circ \tau'$ . Therefore, it is also the case that  $(\psi \circ \tau \circ \sigma_p)(s) = (\psi' \circ \tau' \circ \sigma_p)(s)$  for all  $s \in S$ . Now we must show that  $(p, \tau')$  is a best-response strategy to mechanism  $M'$ . Consider any valid alternative strategy  $(p^*, \tau^*)$ . We aim to show that

$$\mathbb{E}_s y_{s,p^*}(\psi' \circ \tau^* \circ \sigma_{p^*})(s) \leq \mathbb{E}_s y_{s,p}(\psi' \circ \tau' \circ \sigma_p)(s). \quad (1)$$

First, we can assume without loss of generality that  $\tau^*$  always outputs a solution in  $\mathcal{R}$  because  $\psi'$  produces  $\perp$  (and a utility of 0) for all proposals in  $\Psi \setminus \mathcal{R}$ . Then  $\psi' \circ \tau^* = \tau^*$  and, by definition of  $\mathcal{R}$ , we can write  $\tau^* = \psi \circ \hat{\tau}$  for some function  $\hat{\tau}$  from strings to  $\Sigma$ . Then the left hand side of (1) becomes the expected utility of response  $(p^*, \hat{\tau})$  against mechanism  $M = (\Sigma, \psi)$ :

$$\mathbb{E}_s y_{s,p^*}(\psi' \circ \tau^* \circ \sigma_{p^*})(s) = \mathbb{E}_s y_{s,p^*}(\psi \circ \hat{\tau} \circ \sigma_{p^*})(s)$$

whereas the right hand side of (1) is the expected utility of response  $(p, \tau)$  against  $M$ :

$$\mathbb{E}_s y_{s,p}(\psi' \circ \tau' \circ \sigma_p)(s) = \mathbb{E}_s y_{s,p}(\psi \circ \tau \circ \sigma_p)(s).$$

Since  $(p, \tau)$  is a best response for this mechanism, the desired inequality (1) follows.  $\blacktriangleleft$

## 4 Deterministic Mechanisms

In this section, we will consider deterministic single-proposal mechanisms for delegated stochastic probing problems, as defined in Section 3.1. This is in contrast to randomized mechanisms which we will define later in Section 5. We will show that large classes of these problems have constant-factor policies, and therefore constant-factor delegation gaps.

The focus of this section is on Theorem 4.1 and Theorem 4.5, which together give us a general method of constructing competitive delegation policies from certain prophet inequalities and adaptivity gaps. In particular, Corollary 4.4 gives us constant-factor policies for delegated stochastic probing with no outer constraint and an inner constraint which is the intersection of a constant number of matroid, knapsack, and matching constraints. Similarly, Corollary 4.8 gives us constant-factor policies for delegated stochastic probing with any downwards-closed outer constraint and an inner constraint which is the intersection of a constant number of matroids.

### 4.1 Inner Constraint Delegation

We will now consider instances of delegated stochastic probing for which there is no outer constraint. We will then combine the results from this section with Theorem 4.5 to get solutions to delegation problems with both inner and outer constraints.

To simulate the lack of an outer constraint, we will consider instances of delegation for which the outer constraint is the trivial set system in which all subsets of the elements are feasible. For any ground set  $E$  of elements, we will write this trivial set system as  $\mathcal{M}_E^*$ , omitting the subscript when the set of elements  $E$  is clear from context.

► **Theorem 4.1.** *Given an instance  $I = (E, \mathcal{M}^*, \mathcal{M}_{in})$  of delegated stochastic probing without outer constraints, let  $J$  be an instance of the prophet inequality problem with random variables  $X_e$  for all  $e \in E$  and constraint  $\mathcal{M}_{in}$ . If there exists an  $\alpha$ -factor greedy strategy for  $J$  against the almighty adversary, then there exists a deterministic  $\alpha$ -policy for  $I$ . Furthermore, the proof is constructive when given the strategy for  $J$ .*

**Proof.** First, we have by our choice of  $J$  that the expected utility of the prophet in  $J$  is equal to the expected utility of the non-delegating principal in  $I$ . Notice that the principal has no outer constraint, so we can assume without loss of generality that they probe all elements. Then the prophet and non-delegating principal both get exactly

$$\mathbb{E} \max_{T \in \mathcal{M}_{in}} x(T).$$

Now consider the gambler's  $\alpha$ -factor greedy strategy, which consists of some collection  $\mathcal{A} \subseteq 2^{E \times \mathbb{R}_+}$  of “acceptable” sets of outcomes. We will define the delegating principal's policy as follows

$$\mathcal{R} = \{ \{(e, x, y) : (e, x) \in A, y \in \mathbb{R}_+\} : A \in \mathcal{A} \}.$$

Notice that policy  $\mathcal{R}$  is exactly the same as strategy  $\mathcal{A}$ , just translated into the language of delegation.

Now we will show that the utility of the delegating principal with policy  $\mathcal{R}$  is at least the utility of the gambler with greedy strategy  $\mathcal{A}$ . In the prophet inequality, the almighty adversary can order the random variables such that the gambler always gets their least favorite among all maximal acceptable sets (the set is always maximal because the gambler's strategy is greedy). Compare this with delegation, where the agent knows the result of all probed elements as well as the principal's acceptable sets  $\mathcal{R}$ . Since the agent has non-negative utility for all outcomes, we can assume without loss of generality that they will always propose a maximal acceptable set. For every corresponding set of realizations in each problem, the gambler will receive the maximal set in  $\mathcal{A}$  of minimum value and the principal will receive some maximal set in  $\mathcal{R}$ . Since we defined  $\mathcal{R}$  to correspond directly with  $\mathcal{A}$ , the principal's value must be at least as large as the gambler's. This is true of all possible realizations, so  $\mathcal{R}$  must be an  $\alpha$ -policy for  $I$ . ◀

We note that by construction of the principal's policy  $\mathcal{R}$ , this theorem holds even when the principal is unaware of the agent's utility values  $y$ . This is comparable to the reduction in [16] which similarly worked regardless of the principal's knowledge of the agent's utilities.

Unfortunately, applications of this theorem rely on the existence of competitive strategies against the almighty adversary, which is a very strong condition. It is natural to ask whether it's really necessary in the reduction for the adversary to be almighty. We provide some evidence that this is indeed necessary by considering the special case of a 1-uniform inner matroid. In this case, it's easy to construct instances for which the utility of the principal and agent sum to a constant value for all outcomes, i.e.  $X_e + Y_e = c$  for all  $e$  and some constant  $c$ . In such an instance, the agent's goals are directly opposed to the principal's, so the agent will always propose the principal's least favorite acceptable outcome. In the corresponding instance of the prophet inequality, the almighty adversary can guarantee that the gambler chooses their least favorite acceptable outcome, while weaker adversaries (that don't know the realizations of variables) cannot enforce the same guarantee.

Using some known greedy prophet inequalities against the almighty adversary, we get the following corollaries.

► **Corollary 4.2.** *There exist deterministic  $\frac{1}{2}$ -policies for delegated stochastic probing problems with no outer constraint and a 1-uniform inner constraint.*

**Proof.** This follows from the existence of  $\frac{1}{2}$  threshold rules (such as Samuel-Cahn’s median rule [25]) for the 1-uniform prophet inequality against the almighty adversary. ◀

► **Corollary 4.3.** *There exist constant-factor deterministic policies for delegated stochastic probing problems with no outer constraint and three classes of inner constraints. These factors are:  $\frac{1}{4}$  for matroid constraints,  $\frac{1}{2e} \approx 0.1839$  for matching constraints, and  $\frac{3}{2} - \sqrt{2} \approx 0.0857$  for knapsack constraints.*

**Proof.** This corollary is largely based on results from [9]. By combining [9, Theorem 1.8] with [9, Observation 1.6] and optimizing the parameters, we get randomized greedy online contention resolution schemes (OCRS) for three aforementioned constraint systems with the same factors listed above. Then, applying [9, Theorem 1.12], each randomized greedy OCRS corresponds to a randomized greedy prophet inequality against the almighty adversary with the same approximation factor. Since the adversary is almighty, they can predict any randomness in our strategy. Therefore, the randomized strategy is no better than the best deterministic strategy, and there must exist some deterministic strategy achieving the same factor. Finally, we apply our Theorem 4.1 to turn the prophet inequality strategy into a delegation policy with the same factor. ◀

► **Corollary 4.4.** *There exist constant-factor deterministic policies for delegated stochastic probing problems with no outer constraint and an inner constraint that is the intersection of a constant number of matroid, knapsack, and matching constraints.*

**Proof.** We use [9, Corollary 1.13] along with the same reasoning as Corollary 4.3. ◀

We note that it is open whether there exists a  $\frac{1}{2}$ -OCRS for matroids against the almighty adversary [21]. The existence of such an OCRS, if greedy, would imply the existence of  $\frac{1}{2}$ -policy for delegated stochastic probing with a matroid inner constraint and no outer constraint.

Although Corollary 4.2 applies to a model very similar to the distributional delegation model from [16], our principal has the additional power of being able to distinguish between otherwise identical elements by their name alone. However, by observing that Theorem 4.1 turns greedy prophet inequalities that don’t distinguish between identical elements into delegation policies that also don’t distinguish between identical elements, we can derive delegation policies that recover the  $\frac{1}{2}$ -factor guarantee from [16] for their distributional model. We leave the details for Section A.1.

## 4.2 Outer Constraint Delegation

Using the adaptivity gap from Section 2.3, we will now show that there are large classes of delegated stochastic probing problems with nontrivial outer constraints for which the principal can achieve, in expectation, a constant-factor of their non-delegated optimal utility.

► **Theorem 4.5.** *Let  $I = (E, \mathcal{M}_{out}, \mathcal{M}_{in})$  be an instance of delegated stochastic probing. Suppose that, for all  $F \in \mathcal{I}_{out}$ , there exists a deterministic  $\alpha$ -policy for the restriction  $I_F = (F, \mathcal{M}_F^*, \mathcal{M}_{in}|_F)$  of instance  $I$  to  $F$ . Suppose also that the adaptivity gap for weighted rank functions of  $\mathcal{M}_{in}$  on set system  $\mathcal{M}_{out}$  is at least  $\beta$ . Then there exists a deterministic  $\alpha\beta$ -policy for instance  $I$ .*

### 37:12 Delegated Stochastic Probing

**Proof.** Given any set of elements  $F \subseteq E$ , we can write the utility of the non-delegating principal who probes  $F$  as

$$u(F) = \max_{G \subseteq F, G \in \mathcal{I}_{\text{in}}} x(G)$$

and the utility of the delegating principal with policy  $\mathcal{R}$  who probes  $F$  as

$$v_{\mathcal{R}}(F) = x \left( \operatorname{argmax}_{G \subseteq F, \Omega_G \in \mathcal{R}} y(G) \right),$$

where  $\Omega_G \subseteq \Omega$  is the set of outcomes from the probed elements  $G$ .

Notice that for any fixed set of realizations from all random variables,  $u$  is just the weighted rank function of set system  $\mathcal{M}_{\text{in}}$ . Therefore, by the adaptivity gap for such a function over set system  $\mathcal{M}_{\text{out}}$ , there exists a fixed set  $F \in \mathcal{I}_{\text{out}}$  such that

$$\mathbb{E} u(F) \geq \beta \mathbb{E} u(E^*), \quad (2)$$

where  $E^* \in \mathcal{I}_{\text{out}}$  is a random variable representing the optimal set of elements selected by an adaptive non-delegating principal. Notice that expectation is also over the randomness of  $E^*$ .

Now we will consider the same delegation instance with access to only the elements in  $F$ , i.e. instance  $(F, \mathcal{M}_{\text{out}}|F, \mathcal{M}_{\text{in}}|F)$ . Since  $F \in \mathcal{I}_{\text{out}}$ , the outer matroid doesn't restrict probing at all and this instance is equivalent to  $I_F = (F, \mathcal{M}_F^*, \mathcal{M}_{\text{in}}|F)$ . By our assumption, this problem has some  $\alpha$ -approximate delegation policy. Let  $\mathcal{R}$  be one such policy. Then we have

$$\mathbb{E} v_{\mathcal{R}}(F) \geq \alpha \mathbb{E} u(F). \quad (3)$$

Since  $\mathcal{R}$  contains outcomes only from elements in  $F$ , an agent restricted to  $\mathcal{R}$  in the original instance  $I$  has no incentive to probe elements outside of  $F$ . Because  $F \in \mathcal{I}_{\text{out}}$ , the agent can probe all of  $F$ . Therefore, we can assume without loss of generality that an optimal adaptive strategy will choose to probe exactly the elements in  $F$ . Then

$$\mathbb{E} v_{\mathcal{R}}(E_{\mathcal{R}}^*) = \mathbb{E} v_{\mathcal{R}}(F), \quad (4)$$

where  $E_{\mathcal{R}}^* \subseteq E$  is a random variable containing exactly the elements probed by an optimal adaptive agent when restricted to acceptable set  $\mathcal{R}$  in the original instance  $I$ .

Combining (2), (3), and (4), we get the desired inequality:

$$\begin{aligned} \mathbb{E} v_{\mathcal{R}}(E_{\mathcal{R}}^*) &= \mathbb{E} v_{\mathcal{R}}(F) \\ &\geq \alpha \mathbb{E} u(F) \\ &\geq \alpha \beta \mathbb{E} u(E^*). \end{aligned} \quad \blacktriangleleft$$

► **Corollary 4.6.** *There exist deterministic  $\frac{1}{2} \left(1 - \frac{1}{e}\right) \approx 0.3160$ -policies for delegated stochastic probing problems with matroid outer constraints and a 1-uniform inner constraint.*

**Proof.** By Corollary 4.2, there is a  $\frac{1}{2}$ -policy for any instance of delegated stochastic probing with a 1-uniform inner constraint and no outer constraint. Every restriction of our present instance  $I$  to some independent set  $F$  of the outer matroid is of this form.

From [4], we have a  $1 - \frac{1}{e}$  adaptivity gap for stochastic submodular on matroid constraints. Since the weighted rank function of any matroid is submodular, the adaptivity gap of weighted rank functions of the inner 1-uniform matroid constraint on the outer matroid constraint is also  $1 - \frac{1}{e}$ .

Therefore, the conditions of Theorem 4.5 hold with  $\alpha = \frac{1}{2}$  and  $\beta = 1 - \frac{1}{e}$ , and we get the desired factor. ◀

► **Corollary 4.7.** *There exist deterministic  $\frac{1}{4}(1 - \frac{1}{e}) \approx 0.1580$ -policies for delegated stochastic probing problems with matroid outer and inner constraints.*

**Proof.** Similar to Corollary 4.6, we use the  $1 - \frac{1}{e}$  adaptivity gap for submodular functions over matroid constraints along with Corollary 4.3. ◀

► **Corollary 4.8.** *There exist constant-factor deterministic policies for delegated stochastic probing with any downward-closed outer constraint and an inner constraint which is the intersection of a constant number of matroids.*

**Proof.** By [5, Theorem 1.2], we have constant-factor adaptivity gaps for weighted rank functions of the intersection of a constant number of matroids over “prefix-closed” constraints, which include all downward-closed constraints. By Corollary 4.4, we have constant-factor policies for delegated stochastic probing with no outer constraint and an inner constraint which is the intersection of a constant number of matroids. Combining these results with Theorem 4.5, we get the desired constant factors. ◀

## 5 Lottery Mechanisms

One natural generalization of the delegated stochastic probing model defined in section 3.1 is to allow the principal to use randomized mechanisms. For example, one may consider the generalization of single-proposal mechanisms which attaches a probability  $p_R$  to each set of outcomes  $R \subseteq \Omega_{\text{in}}$ , and accepts a proposed set  $R$  with precisely that probability (and accepts the empty set otherwise). More general lotteries (i.e. with non-binary support) are also possible. It’s then natural to ask whether there exist instances for which some randomized policy does better than all deterministic ones. Even further, we can ask whether there exists a randomized policy that strictly outperforms deterministic ones in the worst case. In other words, can randomization give us a strictly better delegation gap?

In this section, we will broadly discuss randomized mechanisms and then consider the special case of delegation with 1-uniform inner constraints and no outer constraints. In this special case, there exist instances for which randomization significantly helps the principal, and there are worst-case instances in which the delegation gap of  $\frac{1}{2}$  is tight for randomized mechanisms as well as deterministic ones. Before getting to these results, we will discuss methods of randomization and then formalize what we mean by a randomized mechanism.

There are two obvious ways that the single-proposal mechanism can be randomized. The first is for the principal to sample a deterministic policy  $\mathcal{R}$  from some distribution and then run the single proposal mechanism defined by  $\mathcal{R}$ . However, noticing that our model of delegation is a Stackelberg game, we can conclude that there always exists a pure optimal strategy for the principal, so this type of randomization doesn’t give the principal any more power.

The second type of randomness is for the policy itself to be a set of acceptable distributions over sets of outcomes (i.e. a menu of lotteries), from which the agent may propose one. The principal then samples a set of outcomes from the proposed lottery. This expands the space of mechanisms, conceivably affording the principal more power in influencing the agent’s behavior. We will focus on these randomized mechanisms for the rest of this section.

► **Definition 5.1.** *A lottery mechanism is a randomized mechanism for delegated stochastic probing consisting of a set  $\mathcal{R}$  of distributions, or lotteries, each with support in  $\Omega_{\text{in}}$ . After the set of acceptable lotteries  $\mathcal{R}$  is selected and announced to the agent, the delegated stochastic probing mechanism proceeds largely the same. The agent probes outcomes  $S$  and proposes*

one of the lotteries  $L \in \mathcal{R}$ . The principal then samples a set of outcomes  $T \sim L$  from that lottery. If  $T \subseteq S$ , then the principal and agent receive  $x(T)$  and  $y(T)$  utility, respectively. Otherwise, they both receive 0 utility.

We note that this sort of mechanism is a generalized single-proposal mechanism in the sense of Section 3.2: Each lottery represents a solution and an agent's expected utility for a lottery represents their utility for that solution. Therefore, Lemma 3.7 applies to lottery mechanisms as well.

## 5.1 Power of Lotteries

The increased power of lottery mechanisms means that for some instances of delegated stochastic probing there exist lottery policies that provide the principal with a better expected utility than the best deterministic policies. In fact, we will show that there are instances for which some lottery policies nearly achieve the principal's non-delegated expected utility, while the best deterministic policies achieve only about half of this value.

First, we will make the observation that it never benefits the principal to declare two lotteries in  $\mathcal{R}$  with identical support but different distributions. This is because the principal knows the utility function of the agent and can predict which lottery the agent will prefer. Therefore, we can assume that for any given support, the principal will declare at most one lottery.

► **Proposition 5.2.** *For all  $0 < \epsilon < 1$ , there exists an instance of delegated stochastic probing for which the best lottery mechanisms achieve  $\frac{2-3\epsilon+2\epsilon^2}{2-\epsilon}$  of the principal's non-delegated expected utility, while the best deterministic mechanisms achieve  $\frac{1}{2-\epsilon}$  of the principal's non-delegated expected utility. As  $\epsilon$  approaches 0, the former approaches 1 while the latter approaches  $\frac{1}{2}$ .*

**Proof.** Consider an instance with elements  $E = \{1, 2\}$ , a 1-uniform matroid inner constraint, no outer constraint, and distributions for elements 1 and 2 as detailed in Table 1.

■ **Table 1** Each row represents a single outcome and contains its name, element  $e$ , utilities  $x$  and  $y$ , and the probability that it is probed from element  $e$ .

Outcome	Element $e$	Utility $x$	Utility $y$	Probability $\mathbb{P}_e[(x, y)]$
$\omega_0$	1	0	0	$1 - \epsilon$
$\omega_1$	1	$1/\epsilon$	$1 - \epsilon$	$\epsilon$
$\omega_2$	2	1	1	1

Since there are no outer constraints we assume that both elements are probed. The non-delegating principal can accept  $\omega_1$  when it appears and accept  $\omega_2$  otherwise. This gives them an expected utility of  $\epsilon/\epsilon + 1 - \epsilon = 2 - \epsilon$ . By enumerating all deterministic policies, we can confirm that the best such policy gives the delegating principal an expected utility of 1. Therefore, the best deterministic policy achieves  $\frac{1}{2-\epsilon}$  of the principal's non-delegated utility.

Now consider a lottery policy with lotteries  $A$  and  $B$  such that  $\mathbb{P}_A[\omega_1] = 1$ ,  $\mathbb{P}_B[\omega_2] = 1 - 2\epsilon$ , and  $\mathbb{P}_B[\omega_0] = 2\epsilon$ . We can quickly verify that this gives the delegating principal an expected utility of  $2 - 3\epsilon + 2\epsilon^2$ . Therefore, at least one lottery policy achieves  $\frac{2-3\epsilon+2\epsilon^2}{2-\epsilon}$  of the principal's non-delegated utility. ◀

Unfortunately, there is good reason not to be too optimistic about the increased power of lottery mechanisms. As we will now show, there also exist instances for which the best lottery policies and the best deterministic policies all achieve approximately half of the principal's

non-delegated expected utility. Since Corollary 4.2 gives us a deterministic  $\frac{1}{2}$ -policy, this tells us that, in the worst case, the factor  $\frac{1}{2}$  is tight even for lottery policies in the special case of no outer constraint and a 1-uniform inner constraint.

► **Proposition 5.3.** *For all  $0 < \epsilon < 1$ , there exists an instance of delegated stochastic probing with a 1-uniform inner constraint and no outer constraint for which the best lottery policies and the best deterministic policies all achieve  $\frac{1}{2-\epsilon}$  of the principal's non-delegated expected utility. As  $\epsilon$  approaches 0, this approaches  $\frac{1}{2}$ .*

**Proof.** Consider an instance with elements  $E = \{1, 2\}$ , a 1-uniform matroid inner constraint, no outer constraint, and distributions for elements 1 and 2 as detailed in Table 2.

■ **Table 2** Each row represents a single outcome and contains its name, element  $e$ , utilities  $x$  and  $y$ , and the probability that it is probed from element  $e$ . Notice that this instance is identical to the one from Table 1 except for the agent's utility for outcome  $\omega_1$ .

Outcome	Element $e$	Utility $x$	Utility $y$	Probability $\mathbb{P}_e[(x, y)]$
$\omega_0$	1	0	0	$1 - \epsilon$
$\omega_1$	1	$1/\epsilon$	0	$\epsilon$
$\omega_2$	2	1	1	1

In the case of ties, we assume that the agent prefers to break ties first in the principal's favor and then arbitrarily among any remaining ties. This assumption serves only to simplify the proof, and can be avoided with careful modifications to the utility table.

The non-delegating principal can accept  $\omega_1 = (1, 1/\epsilon, 0)$  when it appears and accept  $\omega_2 = (2, 1, 1)$  otherwise. This gives them an expected utility of  $\epsilon/\epsilon + 1 - \epsilon = 2 - \epsilon$ . By enumerating all deterministic policies, we can confirm that the best such policy gives the delegating principal an expected utility of 1. Therefore, the best deterministic policy achieves  $\frac{1}{2-\epsilon}$  of the principal's non-delegated utility.

Finding the best menu of lotteries takes slightly more work. Since the inner constraint is 1-uniform, each lottery is supported on singletons as well as the empty set. Recall also that we can restrict attention to menus where no two lotteries have the same support. We claim that we can further restrict attention to menus with exactly two lotteries  $A$  and  $B$ , with  $A$  supported on  $\{\omega_0, \omega_2\}$  and  $B$  supported on  $\{\omega_1, \omega_2\}$ . To see this, observe that:

1. Shifting all probability mass from the empty set to  $\omega_0$  or  $\omega_1$  in any lottery does not affect the agent's utility and can only increase the principal's utility. In the case of tie-breaking, the principal's favorite remaining lottery is no worse than before.
2. If there is a lottery with both  $\omega_0$  and  $\omega_1$  in its support, shifting all probability mass from one of these outcomes to the other does not affect the agent's utility, and in at least one direction this shift of probability mass will make the policy no worse for the principal. Again, in the case of tie-breaking, the principal's favorite remaining lottery is no worse than before.
3. A menu without lottery  $A$  is no better than the same menu with lottery  $A$  for which all probability mass of  $A$  is assigned to  $\omega_0$ . Similarly, a menu without lottery  $B$  is no better than the same menu with lottery  $B$  for which all probability mass of  $B$  is assigned to  $\omega_1$ .

Parametrizing the probability of each outcome, we get:  $\mathbb{P}_A[\omega_2] = a$ ,  $\mathbb{P}_A[\omega_0] = 1 - a$ ,  $\mathbb{P}_B[\omega_2] = b$ , and  $\mathbb{P}_B[\omega_1] = 1 - b$  for some  $a, b \in [0, 1]$ . No matter what the agent probes ( $\{\omega_0, \omega_2\}$  or  $\{\omega_1, \omega_2\}$ ), their favorite lottery is  $B$  if  $b \geq a$  and  $A$  otherwise. If we choose  $b \geq a$ , the delegating principal gets expected utility  $\epsilon(b + (1 - b)/\epsilon) + (1 - \epsilon) \cdot b = 1$ . Otherwise, the principal gets  $\epsilon \cdot a + (1 - \epsilon) \cdot a = a$ , which can be made as large as 1 for  $a = 1$ . Therefore, the best lottery policy achieves  $\frac{1}{2-\epsilon}$  of the principal's non-delegated utility. ◀



## 6 Open Questions

Due to this novel combination of delegation with stochastic probing, we believe that this paper ultimately opens up many more questions than it answers. In this section, we will make some of these questions explicit.

- While we focused on the existence of constant-factor delegation policies regardless of their computational complexity, applying these solutions to practical problems requires some guarantee that they can be easily computed and represented. Are there delegated stochastic probing problems for which constant-factor policies are NP-hard to compute in general? Are there special cases for which constant-factor policies can always be computed in polynomial time?
- In Section 5, we showed that the constant given in Corollary 4.2 is tight. Are the other factors given in Section 4.1 tight? We note that this is related to an open question by [21] about  $\frac{1}{2}$  prophet inequalities on matroids against the almighty adversary.
- Are the constant factors given in Section 4.2 tight? Due to the broad applicability of adaptivity gaps, our method is unlikely to take advantage of special structure that may be present in delegated stochastic probing problems. Therefore, it seems probable that better constants exist, but we make no claim to a conjecture.
- Our model assumes that probing is always zero-cost, so it doesn't generalize the binary model from [16] or the box problem of [27]. It's natural to ask whether we can get constant-factor delegation gaps with probing costs in addition to (or as a replacement for) outer constraints.
- Our model doesn't allow the principal to incentivize the agent with transfers (such as payments), so it's natural to ask how such an extension to the model could improve the principal's worst-case guarantees.
- If the principal is delegating to multiple agents simultaneously, can they get better worst-case guarantees than delegating to a single agent? We note that there are many ways to define this formally. For example, a stronger principal may be able to define different acceptable sets for each agent whereas a weaker principal may be forced to declare one acceptable set for all agents.
- It's not hard to imagine practical applications of stochastic probing for which elements are not independently distributed. Can we get competitive guarantees even in the absence of the independence assumption?

---

## References

- 1 Marek Adamczyk and Michał Włodarczyk. Random order contention resolution schemes. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 790–801. IEEE, 2018.
- 2 Ricardo Alonso and Niko Matouschek. Optimal delegation. *The Review of Economic Studies*, 75(1):259–293, 2008.
- 3 Mark Armstrong and John Vickers. A model of delegated project choice. *Econometrica*, 78(1):213–244, 2010.
- 4 Arash Asadpour and Hamid Nazerzadeh. Maximizing stochastic monotone submodular functions. *Management Science*, 62(8):2374–2391, 2016.
- 5 Domagoj Bradac, Sahil Singla, and Goran Zuzic. (near) optimal adaptivity gaps for stochastic multi-value probing. *arXiv preprint*, 2019. [arXiv:1902.01461](https://arxiv.org/abs/1902.01461).
- 6 Chandra Chekuri, Jan Vondrák, and Rico Zenklusen. Submodular function maximization via the multilinear relaxation and contention resolution schemes. *SIAM Journal on Computing*, 43(6):1831–1879, 2014.

- 7 Ning Chen, Nicole Immorlica, Anna R Karlin, Mohammad Mahdian, and Atri Rudra. Approximating matches made in heaven. In *International Colloquium on Automata, Languages, and Programming*, pages 266–278. Springer, 2009.
- 8 Paul Dütting, Michal Feldman, Thomas Kesselheim, and Brendan Lucier. Prophet inequalities made easy: Stochastic optimization by pricing non-stochastic inputs. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 540–551. IEEE, 2017.
- 9 Moran Feldman, Ola Svensson, and Rico Zenklusen. Online contention resolution schemes. In *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, pages 1014–1033. Society for Industrial and Applied Mathematics, 2016.
- 10 Anupam Gupta and Viswanath Nagarajan. A stochastic probing problem with applications. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 205–216. Springer, 2013.
- 11 Anupam Gupta, Viswanath Nagarajan, and Sahil Singla. Algorithms and adaptivity gaps for stochastic probing. In *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, pages 1731–1747. SIAM, 2016.
- 12 Mohammad Taghi Hajiaghayi, Robert Kleinberg, and Tuomas Sandholm. Automated online mechanism design and prophet inequalities. In *AAAI*, volume 7, pages 58–65, 2007.
- 13 Bengt Holmstrom. On the theory of delegation. Technical report, Discussion Paper, 1980.
- 14 Bengt Robert Holmstrom. *On Incentives and Control in Organizations*. PhD thesis, Stanford University, 1978.
- 15 Ali Khodabakhsh, Yuanzhe Liu, Emmanouil Pountourakis, Sam Taggart, and Yichi Zhang. Threshold policies for delegation. *working paper*, 2020.
- 16 Jon Kleinberg and Robert Kleinberg. Delegated search approximates efficient search. In *Proceedings of the 2018 ACM Conference on Economics and Computation*, pages 287–302, 2018.
- 17 Robert Kleinberg and Seth Matthew Weinberg. Matroid prophet inequalities. In *Proceedings of the forty-fourth annual ACM Symposium on Theory of Computing*, pages 123–136. ACM, 2012.
- 18 Eugen Kováč and Tymofiy Mylovanov. Stochastic mechanisms in settings without monetary transfers: The regular case. *Journal of Economic Theory*, 144(4):1373–1395, 2009.
- 19 Ulrich Krengel and Louis Sucheston. Semiamarts and finite values. *Bulletin of the American Mathematical Society*, 83(4):745–747, 1977.
- 20 Ulrich Krengel and Louis Sucheston. On semiamarts, amarts, and processes with finite value. *Probability on Banach spaces*, 4:197–266, 1978.
- 21 Euiwoong Lee and Sahil Singla. Optimal online contention resolution schemes via ex-ante prophet inequalities. In *26th Annual European Symposium on Algorithms (ESA 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.
- 22 Brendan Lucier. An economic view of prophet inequalities. *ACM SIGecom Exchanges*, 16(1):24–47, 2017.
- 23 Nahum D Melumad and Toshiyuki Shibano. Communication in settings with no transfers. *The RAND Journal of Economics*, pages 173–198, 1991.
- 24 Tim Roughgarden. *Twenty lectures on algorithmic game theory*. Cambridge University Press, 2016.
- 25 Ester Samuel-Cahn et al. Comparison of threshold stop rules and maximum for independent nonnegative random variables. *the Annals of Probability*, 12(4):1213–1216, 1984.
- 26 Sahil Singla. *Combinatorial Optimization Under Uncertainty: Probing and Stopping-Time Algorithms*. PhD thesis, PhD thesis, Carnegie Mellon University, 2018.
- 27 Martin L Weitzman. Optimal search for the best alternative. *Econometrica: Journal of the Econometric Society*, pages 641–654, 1979.

## A

 Appendix

### A.1 Symmetric Delegation Policies

While our model is not a direct generalization of the distributional model used by Kleinberg and Kleinberg, we can obtain a generalization by considering delegated stochastic probing with a restricted class of policies, which we call symmetric policies. Given this variant, we can recover the  $\frac{1}{2}$  factor that they obtained. First, we need to define some notation and terminology.

Given any object  $X$  (such as a set, tuple, or recursive combination of the two) containing atomic elements  $E$ , we can consider the operation of taking two elements  $e_1, e_2 \in E$  and swapping all instances of  $e_1$  and  $e_2$  in  $X$ . More generally, for any permutation  $\pi$  of elements in  $E$ , we can consider rewriting all elements  $e$  to  $\pi(e)$  simultaneously. We will denote the object obtained from this operation as  $X[E \rightarrow \pi(E)]$ .

► **Definition A.1.** Fix an instance of delegated stochastic probing with elements  $E$ , outer constraint  $\mathcal{M}_{out}$ , and inner constraint  $\mathcal{M}_{in}$ . We say that a subset of elements  $F \subseteq E$  are symmetric if  $\mu_e = \mu_f$  for all  $e, f \in F$  and for all permutations  $\pi$  on  $F$  we have that  $\mathcal{M}_{in}[F \rightarrow \pi(F)] = \mathcal{M}_{in}$  and  $\mathcal{M}_{out}[F \rightarrow \pi(F)] = \mathcal{M}_{out}$ .

► **Definition A.2.** Fix an instance of delegated stochastic probing with elements  $E$ , outer constraint  $\mathcal{M}_{out}$ , and inner constraint  $\mathcal{M}_{in}$ . We say that a policy  $\mathcal{R}$  is symmetric if  $\mathcal{R}[F \rightarrow \pi(F)] = \mathcal{R}$  for all symmetric sets of elements  $F \subseteq E$  and all permutations  $\pi$  on  $F$ .

Intuitively, symmetric elements are ones which are identical in everything except name. Then symmetric policies are ones that don't distinguish between symmetric elements. Using this intuition, we will now consider the problem of delegated stochastic probing with  $k$  identically distributed elements  $E$ , a 1-uniform inner constraint, and no outer constraint. Given any such instance, it's easy to see that all elements  $E$  are symmetric. Notice the similarity between such an instance and the distributional model. The only difference is that our principal has the power to distinguish between outcomes sampled from different elements. However, if the principal is restricted to symmetric policies, then their policy cannot distinguish between different elements, so it must characterize acceptable outcomes based only on their  $(x, y)$  utility. This is equivalent to the distributional model.

There are also natural definitions of symmetric elements and strategies in the prophet inequality problem.

► **Definition A.3.** Fix an instance of the prophet inequality problem with elements  $E$  and feasibility constraint  $\mathcal{M}$ . We say that a subset of elements  $F \subseteq E$  are symmetric if  $X_e$  and  $X_f$  are identically distributed for all  $e, f \in F$  and for all permutations  $\pi$  on  $F$  we have that  $\mathcal{M}[F \rightarrow \pi(F)] = \mathcal{M}$ .

► **Definition A.4.** Fix an instance of the prophet inequality problem with elements  $E$  and feasibility constraint  $\mathcal{M}$ . We say that a strategy  $\mathcal{A}$  is symmetric if  $\mathcal{A}[F \rightarrow \pi(F)] = \mathcal{A}$  for all symmetric sets of elements  $F \subseteq E$  and all permutations  $\pi$  on  $F$ .

Given these definitions, we will show that Theorem 4.1 actually transforms symmetric greedy prophet inequalities against the almighty adversary into symmetric delegation policies. This is stated formally in Proposition A.5.

► **Proposition A.5.** *Given an instance  $I = (E, \mathcal{M}^*, \mathcal{M}_{in})$  of delegated stochastic probing without outer constraints, let  $J$  be an instance of the prophet inequality problem with random variables  $X_e$  for all  $e \in E$  and constraint  $\mathcal{M}_{in}$ . If there exists a symmetric  $\alpha$ -factor greedy strategy for  $J$  against the almighty adversary, then there exists a symmetric deterministic  $\alpha$ -policy for  $I$ . Furthermore, the proof is constructive when given the strategy for  $J$ .*

**Proof.** The proof is identical to the proof of Theorem 4.1, but we observe that the greedy strategy  $\mathcal{A}$  for prophet inequality problem  $J$  is symmetric, so the policy  $\mathcal{R}$  derived from  $\mathcal{A}$  must also be symmetric by construction. ◀

Since the  $\frac{1}{2}$  prophet inequality used in Corollary 4.2 is a threshold policy, it must be symmetric. Therefore, we have a symmetric  $\frac{1}{2}$ -policy for delegated stochastic probing problems with no outer constraint and a 1-uniform inner constraint. This recovers a  $\frac{1}{2}$ -factor for the distributional model of [16], as well as for the slight generalization of this model with multiple distributions and a separate cardinality constraint for each one.



# Explicit SoS Lower Bounds from High-Dimensional Expanders

**Irit Dinur**

Weizmann Institute of Science, Rehovot, Israel  
irit.dinur@weizmann.ac.il

**Yuval Filmus** 

Technion - Israel Institute of Technology, Haifa, Israel  
yuvalfi@cs.technion.ac.il

**Prahladh Harsha** 

Tata Institute of Fundamental Research, Mumbai, India  
prahladh@tifr.res.in

**Madhur Tulsiani**

Toyota Technological Institute at Chicago, IL, USA  
madhurt@ttic.edu

---

## Abstract

We construct an explicit and structured family of 3XOR instances which is hard for  $O(\sqrt{\log n})$  levels of the Sum-of-Squares hierarchy. In contrast to earlier constructions, which involve a random component, our systems are highly structured and can be constructed explicitly in deterministic polynomial time.

Our construction is based on the high-dimensional expanders devised by Lubotzky, Samuels and Vishne, known as LSV complexes or Ramanujan complexes, and our analysis is based on two notions of expansion for these complexes: cosystolic expansion, and a local isoperimetric inequality due to Gromov.

Our construction offers an interesting contrast to the recent work of Alev, Jeronimo and the last author (FOCS 2019). They showed that 3XOR instances in which the variables correspond to vertices in a high-dimensional expander are easy to solve. In contrast, in our instances the variables correspond to the edges of the complex.

**2012 ACM Subject Classification** Theory of computation → Semidefinite programming; Theory of computation → Expander graphs and randomness extractors; Theory of computation → Proof complexity

**Keywords and phrases** High-dimensional expanders, sum-of-squares, integrality gaps

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.38

**Related Version** A full version of the paper is available at [arXiv:2009.05218](https://arxiv.org/abs/2009.05218).

**Funding** *Irit Dinur*: ERC-CoG grant number 772839.

*Yuval Filmus*: European Union’s Horizon 2020 research and innovation programme under grant agreement No 802020-ERC-HARMONIC.

*Prahladh Harsha*: Department of Atomic Energy, Government of India, under project no. RTI4001 and the Swarnajayanti fellowship, Department of Science and Technology, Government of India.

*Madhur Tulsiani*: NSF grant CCF-1816372

**Acknowledgements** Part of this work was done when the authors were visiting the Simons Institute of Theory of Computing, Berkeley for the 2019 summer cluster on “Error-Correcting Codes and High-Dimensional Expansion”. We thank the Simons Institute for their kind hospitality.



© Irit Dinur, Yuval Filmus, Prahladh Harsha, and Madhur Tulsiani;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 38; pp. 38:1–38:16

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

## 1 Introduction

We describe a new family of instances of 3XOR, based on high-dimensional expanders, that are hard for the Sum-of-Squares (SoS) hierarchy of semidefinite programming relaxations, which is the most powerful algorithmic framework known for optimizing over constraint satisfaction problems. Unlike previous constructions of 3XOR hard instances for SoS, our construction is explicit, as it is based on the explicit construction of high-dimensional expanders due to Lubotzky, Samuels and Vishne [34, 35], which we refer to henceforth as LSV complexes.

► **Theorem 1.1.** *There exists a constant  $\mu \in (0, 1)$  and an infinite family of 3XOR instances on  $n$  variables, constructible in deterministic polynomial time, satisfying the following:*

- *No assignment satisfies more than  $1 - \mu$  fraction of the constraints.*
- *Relaxations obtained by  $O(\sqrt{\log n})$  levels of the SoS hierarchy fail to refute the instances.*

We remark that the result in the above theorem differs from the previous results for random instances of 3XOR, proved by Grigoriev [20] and Schoenebeck [36], in two ways.

- While random instances are known to be hard for  $\Omega(n)$  levels of the SoS hierarchy, the above theorem only gives a gap for  $\Omega(\sqrt{\log n})$  levels.
- Our instances on the LSV complexes exhibit an integrality gap of  $1 - \mu$  vs. 1, while random instances exhibit a gap of  $1/2 + \varepsilon$  vs. 1. However, our construction can also be combined with reductions in the SoS hierarchy [37] hierarchy, reductions can be used to obtain explicit 3XOR instances with a gap of  $1/2 + \varepsilon$  vs.  $1 - \varepsilon$  for any  $\varepsilon > 0$ . Indeed, this yields explicit hard instances with optimal gaps for all approximation resistant predicates based on pairwise independent subgroups [10].

### Structured instances from high-dimensional expanders

High-dimensional expanders (HDXs) are a high-dimensional analog of expander graphs. In recent years they have found a variety of applications in theoretical computer science, such as efficient CSP optimization [3], improved sampling algorithms [6, 4, 5, 11], quantum LDPC codes [17, 30], novel lattice constructions [29], direct sum testing [19], and others. Explicit constructions of HDXs have also led to improved list-decoding algorithms [13, 2] and to sparser agreement tests [14, 12]. In this work, we show how these explicit constructions can be used to construct explicit hard instances for SoS.

High-dimensional expanders are bounded-degree (hyper)graphs (or rather, simplicial complexes) with certain expansion properties. A simplicial complex is a non-empty collection of down-closed sets. Given a simplicial complex  $X$ , we will refer by  $X(i)$  the family of all  $i$ -dimensional sets in  $X$  (i.e., sets of size  $i + 1$ ). The dimension of the simplicial complex  $X$  is the maximal dimension of any set in it. It will be convenient to refer to the sets of dimension 0, 1, 2, 3 as vertices, edges, triangles, tetrahedra, respectively. Thus, a graph  $G = (V, E)$  is a 1-dimensional complex, while in this work we will be using complexes of dimension at least 2. Given a 2-dimensional complex  $X = (X(0), X(1), X(2))$ , there are two natural ways to construct a 3XOR instance based on  $X$  – a vertex-variable construction and an edge-variable construction. Let  $\beta: X(2) \rightarrow \mathbb{F}_2$  be any  $\mathbb{F}_2$ -valued function on the set  $X(2)$  of triangles.

**Vertex-variable construction:** The 3XOR instance corresponding to  $(X, \beta)$  consists of the following constraints:  $x_u + x_v + x_w = \beta_{\{u,v,w\}}$  for each  $\{u, v, w\} \in X(2)$ .

**Edge-variable construction:** The 3XOR instance corresponding to  $(X, \beta)$  consists of the following constraints:  $x_{\{u,v\}} + x_{\{v,w\}} + x_{\{w,u\}} = \beta_{\{u,v,w\}}$  for each  $\{u, v, w\} \in X(2)$ .



The vertex-variable construction whose underlying structure is a high-dimensional expander has been studied by Alev, Jeronimo and the last author [3]. They gave an efficient algorithm for approximating vertex-variable constraint satisfaction problems (not necessarily 3XOR) on an underlying high-dimensional expander. Their result is a generalization to higher dimensions of the corresponding result for graphs that “CSPs are easy on expanders” [7, 24]. They prove this by showing that certain types of random walks on vertices converge very fast on high-dimensional expanders. However, the same analysis fails to show a similar result for the edge-variable construction, as the corresponding random walk on edges of a high-dimensional expander does not mix. Our work shows that this difference isn’t just a technical limitation of their analysis; it is inherent. The edge-variable variant is truly hard, at least for SoS. This demonstrates an interesting subtlety in the structure of high-dimensional expanders, and how it relates to optimization.

To understand our edge-variable construction better, it will be convenient to set up some notation. Let  $C^i$  denote the set of all  $\mathbb{F}_2$ -valued functions on  $X(i)$ . For each  $0 \leq i < d$ , consider the operator  $\delta_i: C^i \rightarrow C^{i+1}$  defined as follows:

$$\delta_i f(s) := \sum_{u \in s} f(s - \{u\}).$$

This is usually referred to as the coboundary operator. Let  $B^i$  be the image of  $\delta_{i-1}$ , and let  $Z^i$  be the kernel of  $\delta_i$ . Clearly,  $B^i, Z^i \subseteq C^i$ . Furthermore, it is not hard to see that  $B^i \subseteq Z^i$ . It easily follows from the definitions that the edge-variable construction corresponding to  $(X, \beta)$  is a satisfiable instance iff  $\beta \in B^2$ .

Typically, soundness of SoS-hard instances is proved by choosing  $\beta$  at random from  $C^2$ . In contrast, we construct our explicit instances by choosing the function  $\beta$  more carefully, and relying on a certain type of expansion property of the complex. Recall that  $B^2 \subset Z^2$ , and the instance is satisfiable iff  $\beta \in B^2$ . Complexes for which  $B^2 = Z^2$  are said to have trivial second cohomology. We will be working with complexes with non-trivial second cohomology, i.e.,  $B^2 \neq Z^2$ . This lets us choose a  $\beta \in Z^2 \setminus B^2$  to prove soundness. It is known that the explicit constructions of HDXs due to Lubotzky, Samuels and Vishne [34, 35] have non-trivial second cohomology.<sup>1</sup> In fact, these complexes have the stronger property (due to a theorem of Evra and Kaufman [16]) that all  $\beta \in Z^2 \setminus B^2$  are not only not in  $B^2$ , but in fact far from any function in  $B^2$ . This latter property follows from the cosystolic expansion of the complex, and forms the basis for the soundness of our instances.

How do we prove the completeness of our instance, namely, that SoS fails to detect that it is a negative instance? The LSV construction is a quotient of the so-called affine building which is, from a topological point of view, a simple “Euclidean-like” object with trivial cohomologies. The hardness of our instance comes from the inherent difference between the LSV complex and the building, which cannot be seen through local balls whose radius is at most the injectivity radius of the complex, in our case  $\Theta(\log n)$ . Locally, the LSV quotient is isomorphic to the building. However, unlike the building, the LSV complex is a quotient with non-trivial cohomologies. The hardness comes from the fact that local views cannot capture the cohomology, which is a global property. Given this observation, the proof of completeness can be carried out following the argument of Ben-Sasson and Wigderson [8] that any short resolution proof is narrow, and Grigoriev [20] and Schoenebeck [36]’s transformation from resolution lower bounds to SoS lower bounds.

<sup>1</sup> More accurately, their construction depends on the group defining the quotient. They show that a certain choice of groups yields non-trivial second cohomology.

Technically, we rely on two very different types of expansion or isoperimetry. In our proof of completeness, we rely on an isoperimetric inequality called Gromov’s filling inequality, that says that balls are essentially the objects with smallest boundary in any CAT(0) space (a class of spaces that includes both Euclidean spaces and the affine building). In our proof of soundness, we rely on the cosystolic expansion of the LSV complex, as proven by Evra and Kaufman [16], which implies that any non-trivial element in the cohomology has constant weight. Both of these statements are related to expansion, yet they are distinct from other notions of expansion used in previous SoS lower bounds. Interestingly, both notions are natural generalizations of edge-expansion to higher dimensions. Isoperimetric expansion is a classical notion asking for the smallest possible boundary of a body with certain volume. In graphs, it is common to interpret this notion as the edge-expansion, bounding the smallest possible number of edges leaving a set, relative to its size. Moving to higher dimensions, there are several nonequivalent [23] ways to generalize edge-expansion, most notably a spectral variant and a topological variant. The topological variant is the one we require for our soundness analysis. This type of expansion is an extension of the notion of coboundary expansion first defined by Linial-Meshulam [32] and then independently by Gromov [22]. This is a subtle notion that is related to the local-testability of the cocycle space, see [28].

### Relation to previous SoS gap constructions

All previous constructions of hard instances for SoS can be viewed in the vertex/edge-variable framework (typically vertex-variable). To the best of our knowledge, all known hard instances, proving inapproximability in the SoS hierarchy, are *random* instances; either both the complex  $X$  and the function  $\beta$  are random, or just the function  $\beta$  is random. The proof of SoS hardness of these random instances relies on very strong expansion of the underlying complex [36] or on certain pseudorandom properties [31], both of which are not yet known to be explicitly constructible. Moreover, the randomness in the choice of the  $\beta$  specifying the right-hand sides of the equations in these constructions, is used for a union bound over all (exponentially many) assignments to the variables, and such arguments are often difficult to derandomize.

On the other hand, explicit hard instances for SoS are known in proof complexity (e.g., Tseitin tautologies on expanders). However, these instances are only minimally unsatisfiable, and transforming them to an integrality gap instance requires a highly *non-local* reduction (such as the PCP theorem). While SoS gap instances can easily be combined with local reductions, this is not true for non-local ones.

In contrast to the above, our integrality gap instances are “anti-random”. They are very structured and easily distinguishable from random instances. For example, all balls around a vertex up to some radius are identical and have very specific structure. Naturally, the typical analysis that works for random instances cannot work here. For example, soundness for random instances is based on choosing a random  $\beta$  and using a union-bound argument to show that with high probability, every solution violates nearly half of the constraints. In contrast, for us, a random  $\beta$  is not a good choice because the local structure will quickly detect local contradictions, ruining the completeness altogether.

### Open directions

Our construction of explicit hard SoS instances based on HDXs begs several questions, some of which we discuss below.

**Improved soundness** Our construction yields 3XOR hard instances which are at most  $(1 - \mu)$ -satisfiable, owing to the cosystolic expansion of the underlying HDX (more precisely,  $\text{CoSys}^2(X) \geq \mu$ , see Section 2.2 for the definition of  $\text{CoSys}^2$ ). Coupled with reductions

in the SoS hierarchy [37], this yields 3XOR hard instances which are at most  $(1/2 + \varepsilon)$ -satisfiable for every  $\varepsilon \in (0, 1)$ . Can we obtain such a result directly from the HDX construction (bypassing reductions), say by constructing HDXs which satisfy  $\text{CoSys}^2(X) \geq 1/2 - \varepsilon$ ? In addition to maintaining the HDX structure, bypassing reductions would also allow for perfect completeness, which is lost while using NP-hardness reductions.

**Fooling more levels of the SoS hierarchy** Our hard instances fool only  $O(\sqrt{\log n})$  levels of the SoS hierarchy, as our argument is based on the injectivity radius of the complexes, which is  $O(\log n)$ , and we suffer a further square-root loss due to the use of Gromov's isoperimetry inequality. It is possible that a much stronger lower bound holds for these instances. Can one construct explicit hard instances that fool linearly many levels of the SoS hierarchy?

**HDX dimension and CSP definition** We find the contrast between the vertex-variable and edge-variable constructions baffling: while the vertex-variable construction is easy, our result demonstrates the hardness of the edge-variable construction. As we go to higher dimensions of HDX, there are more ways to define CSPs. Which of these are easy and which are hard?

## 2 Preliminaries

### 2.1 The Sum-of-Squares Hierarchy

The sum-of-squares hierarchy<sup>2</sup> provides a hierarchy of semidefinite programming (SDP) relaxations, for various combinatorial optimization problems. Figure 1 describes the relaxation given by  $t$  levels of the hierarchy for an instance  $\mathcal{I}$  of 3XOR in  $n$  variables, with  $m$  constraints of the form  $x_{i_1} + x_{i_2} + x_{i_3} = \beta_{i_1 i_2 i_3}$  over  $\mathbb{F}_2$ . We also use  $\mathcal{I}$  to denote the set of all tuples  $\{i_1, i_2, i_3\}$  present as constraints. A solution to the relaxation is specified by a collection of unit vectors  $\{\mathbf{u}_S\}_{S \subseteq [n], |S| \leq t}$ , satisfying the constraints in the program. The objective equals the fraction of constraints “satisfied” by the SDP solution. To prove a lower bound on the

$$\begin{array}{ll} \text{maximize} & \frac{1}{2} + \frac{1}{2m} \cdot \sum_{\{i_1, i_2, i_3\} \in \mathcal{I}} (-1)^{\beta_{i_1 i_2 i_3}} \cdot \langle \mathbf{u}_{\{i_1, i_2, i_3\}}, \mathbf{u}_\emptyset \rangle \\ \text{subject to} & \langle \mathbf{u}_{S_1}, \mathbf{u}_{S_2} \rangle = \langle \mathbf{u}_{S_3}, \mathbf{u}_{S_4} \rangle \quad \forall S_1 \Delta S_2 = S_3 \Delta S_4, |S_1|, \dots, |S_4| \leq t \\ & \|\mathbf{u}_S\| = 1 \quad \forall S, |S| \leq t \end{array}$$

■ **Figure 1** Relaxation for 3XOR given by  $t$  levels of the SoS hierarchy.

value of the SDP relaxation, we will use the following result, which shows the existence of vectors  $\mathbf{u}_S$  yielding an objective value of 1, when the given system of XOR constraints does not have any “low-width” refutations. Formally, we consider a system called XOR-resolution, where the only rule allows us to combine two equations  $\ell_1 = b_1$  and  $\ell_2 = b_2$  to derive the equation  $\ell_1 + \ell_2 = b_1 + b_2$ . A refutation is a derivation of  $0 = 1$ . The width of a refutation is the maximum number of variables in any equation used in the refutation. We include a proof of the following lemma in Appendix A.

<sup>2</sup> For more on Sum-of-Squares, see the recent monograph by Fleming, Kothari and Pitassi [18].

► **Lemma 2.1** ([36, Lemma 13], [37, Theorem 4.2]). *Let  $\Lambda$  be a system of equations in  $n$  variables over  $\mathbb{F}_2$ , which does not admit any refutations of width at most  $2t$ . Then there exist vectors  $\{\mathbf{u}_S\}_{S \subseteq [n], |S| \leq t}$  satisfying the constraints in Figure 1, such that for all equations  $\sum_{i \in T} x_i = b_T$  in  $\Lambda$  with  $|T| \leq t$ , we have  $\langle \mathbf{u}_T, \mathbf{u}_\emptyset \rangle = (-1)^{b_T}$ .*

## 2.2 Simplicial Complexes

A simplicial complex  $X$  is a non-empty collection of sets (known as faces) which is closed downwards. The  $i$ -dimensional faces  $X(i)$  are all sets of size  $i + 1$ . The dimension of the complex is the maximal dimension of a face. Faces of that dimension are known as facets. Faces of dimensions 0, 1, 2, 3 are called vertices, edges, triangles, and tetrahedra, respectively.

Graphs are 1-dimensional simplicial complexes. The skeleton of a simplicial complex is the graph obtained by retaining only faces of dimension at most 1.

**Links.** Let  $X$  be a  $d$ -dimensional simplicial complex. The link  $X_s$  of a face  $s \in X(i)$  is a simplicial complex of dimension  $d - (i + 1)$  given by  $X_s(j) := \{t : s \cup t \in X(j + i + 1)\}$ . In other words,  $X_s$  contains all faces in  $X$  which contain  $s$ , with  $s$  itself removed.

**Balls.** Let  $X$  be a simplicial complex. A ball of radius  $r$  around a vertex  $v$  is the subcomplex induced by all vertices at distance at most  $r$  from  $v$ , as measured on the skeleton of  $X$ . That is, the subcomplex contains a face of  $X$  if it contains all the vertices of the face.

**Covering map.** A *covering map* from a simplicial complex  $Y$  to a simplicial complex  $X$  is a *surjective* map  $\psi : Y(0) \rightarrow X(0)$  from the vertices of  $Y$  to  $X$  such that for every  $k \leq \dim Y$  the image of every  $k$ -face  $\{v_0, \dots, v_k\} \in Y(k)$  is a  $k$ -face  $\{\psi(v_0), \dots, \psi(v_k)\} \in X(k)$ . We then say that  $X$  is *covered* by  $Y$ .

**Chains.** Fix a  $d$ -dimensional simplicial complex  $X$ . Let  $C^i := C^i(X, \mathbb{F}_2)$  be the set of all functions from  $X(i)$  to  $\mathbb{F}_2$ . Elements of  $C^i$  are also known as  $i$ -chains.

For an  $i$ -chain  $f$ , we define  $|f|$  to be the number of non-zero elements in  $f$ . For two  $i$ -chains  $f$  and  $g$ , we define the distance between  $f$  and  $g$  to be  $\text{dist}(f, g) := |f + g|$ .

**Inner product.** For  $f, f' \in C^i$ , let us denote by  $\langle f, f' \rangle_i$  the following sum modulo 2:

$$\langle f, f' \rangle_i := \sum_{s \in X(i)} f(s) f'(s).$$

This is not an inner product in the usual sense as we are working over a field of non-zero characteristic, but it is convenient notation. We will usually drop the subscript  $i$ .

**Dual space.** Given any subspace  $V \subset C^i$ , the dual of  $V$  (under  $\langle \cdot, \cdot \rangle_i$ ) is defined as:

$$V^\perp := \{f \in C^i \mid \text{for all } g \in V, \langle f, g \rangle_i = 0\}.$$

**Boundaries, Cycles, Homology.** The boundary operator  $\partial_i : C^i \rightarrow C^{i-1}$  is given by

$$\partial_i f(s) := \sum_{t \in X(i) : t \supset s} f(t).$$

It gives rise to boundaries  $B_i$  and cycles  $Z_i$ :

$$B_i := \text{im } \partial_{i+1}, \quad Z_i := \ker \partial_i.$$

In the case of graphs,  $Z_1$  consists of all sums of cycles (in the usual sense).

The coboundary operator  $\delta_i: C^i \rightarrow C^{i+1}$ , which is the adjoint of the boundary operator, is given by

$$\delta_i f(t) := \sum_{s \in X(i): s \subset t} f(s) = \sum_{u \in t} f(t - \{u\}).$$

It gives rise to coboundaries and cocycles:

$$B^i := \text{im } \delta_{i-1}, \quad Z^i := \ker \delta_i.$$

We will usually drop the subscript  $i$  when invoking  $\partial, \delta$ .

It is easy to see that  $B_i \subset Z_i$  (every boundary is a cycle) and  $B^i \subset Z^i$  (every coboundary is a cocycle). For example, in a 2-dimensional complex, the boundary of every triangle is a cycle. We call such cycles *trivial cycles*. Modding out by trivial cycles and cocycles, we obtain the homology and cohomology spaces

$$H_i := Z_i / B_i, \quad H^i := Z^i / B^i.$$

The dimensions of these spaces (which are identical) measure the number of “holes” in a particular dimension. Nice complexes (such as the buildings considered below) have no holes.

The following claim shows that the coboundary operator is the adjoint of the boundary operator.

▷ **Claim 2.2.** Let  $f \in C^i, g \in C^{i-1}$ . Then  $\langle f, \delta g \rangle_i = \langle \partial f, g \rangle_{i-1}$ .

Proof.

$$\begin{aligned} \langle f, \delta_{i-1} g \rangle_i &= \sum_{t \in X(i)} f(t) \cdot \delta_{i-1} g(t) = \sum_{t \in X(i)} f(t) \cdot \left( \sum_{s \in X(i-1): s \subset t} g(s) \right) \\ &= \sum_{s \in X(i-1)} \left( \sum_{t \in X(i): t \supset s} f(t) \right) \cdot g(s) = \langle \partial_i f, g \rangle_{i-1}. \end{aligned} \quad \triangleleft$$

The following two claims show that the dimensions of homology and cohomology spaces are identical.

▷ **Claim 2.3.**  $Z_i = (B^i)^\perp, \quad Z^i = (B_i)^\perp$ .

Proof.  $Z_i = \ker \partial_i = \ker \delta_{i-1}^* = (\text{im } \delta_{i-1})^\perp = (B^i)^\perp$ . ◁

▷ **Claim 2.4.**  $\dim H_i = \dim H^i$

Proof.

$$\begin{aligned} \dim H_i &= \dim Z_i - \dim B_i \\ &= \dim C^i - \dim B^i - \dim B_i && \text{[By Claim 2.3]} \\ &= \dim Z^i - \dim B^i && \text{[By Claim 2.3]} \\ &= \dim H^i. \end{aligned} \quad \triangleleft$$

**Cosystoles.** We define, following Evra and Kaufman [16, Definition 2.14], the  $i$ -cosystole of a complex  $X$  to be the minimal (fractional) size of  $f \in Z^i \setminus B^i$ ,

$$\text{CoSys}^i(X) := \min_{f \in Z^i \setminus B^i} |f|/|X(i)|.$$

## 2.3 The Building $\mathcal{B}^{(d+1)}$

The infinite  $k$ -regular tree is the unique connected  $k$ -regular graph without cycles. Affine buildings are higher-dimensional analogs of the infinite  $k$ -regular tree. For  $d = 1$ , the one-dimensional affine building  $\mathcal{B}^{(1)}$  is the  $k$ -regular tree. For higher dimensions they are *regular* in the sense that all vertex links are bounded and identical in structure, they are connected and contractible,<sup>3</sup> and so have vanishing cohomologies, that is, the cohomology spaces  $H^1, \dots, H^{d-1}$  are trivial, where  $d$  is the dimension.

We won't describe  $\mathcal{B}^{(d+1)}$  any further; the interested reader can check [26, 1]. A crucial property of  $\mathcal{B}^{(d+1)}$  which we will need in the sequel is its being a CAT(0) space,<sup>4</sup> which is a geometric definition capturing non-positive curvature; see [9] for more information. The property of being CAT(0) has the following implication, due to Gromov [21, 25, 38]:

► **Theorem 2.5** (Gromov's filling inequality for CAT(0) spaces). *For every cycle  $f \in Z_1$  there is a filling  $g \in C_2$  such that  $f = \partial g$  and  $|g| = O(|f|^2)$ .*

Gromov's filling inequality is an isoperimetric inequality. It generalizes the classic isoperimetric inequality in the plane, which states that any simple closed curve of length  $L$  encloses a region whose area is at most  $L^2/4\pi$ .

The isoperimetric inequality in the plane can be stated in an equivalent way: the boundary of any *bounded* region of area  $A$  is a curve whose length is at least  $\sqrt{4\pi A}$ . This inequality fails for unbounded regions, which could have infinite area but finite boundary (for example, consider the complement of a circle). In the same way, Gromov's inequality doesn't imply that each  $g \in C_2$  satisfies  $|\partial g| = \Omega(\sqrt{|g|})$ . Rather, we have to replace  $|g|$  with  $\min_{h \in Z_2} |g + h|$ .

Gromov's filling inequality also applies to  $i$ -chains, with an exponent of  $i + 1$ , but we will only need the case  $i = 1$ .

In the sequel, we will apply Gromov's filling inequality not to the building itself, but rather to balls in the building. The CAT(0) property almost immediately implies that a ball in a CAT(0) space is itself CAT(0) [9, Exercise II.1.6]. Furthermore, it is well-known that CAT(0) spaces are contractible, and so have vanishing homologies.

► **Lemma 2.6.** *Balls in  $\mathcal{B}^{(d+1)}$  have vanishing homologies and satisfy Gromov's filling inequality.*

## 2.4 The LSV quotient

Whereas the affine building is an infinite simplicial complex, Lubotzky, Samuels and Vishne constructed a growing family of finite complexes that are obtained from quotients of the affine building. These quotients have a growing number of vertices, and locally, in a ball

<sup>3</sup> A complex is contractible, roughly speaking, if it can be continuously deformed to a point (technically, it is homotopy-equivalent to a point). Since (co)homologies are preserved by such deformations, all (co)homologies of a contractible complex vanish.

<sup>4</sup> A space is CAT(0) if for every triangle  $x, y, z$ , the distance between  $x$  and the midpoint of  $y, z$  is at most the corresponding distance in a congruent triangle in Euclidean space.

around each vertex, the complex is isomorphic to the affine building. Moreover, they gave a very explicit algorithm for constructing these complexes by first constructing a Cayley graph with an explicit set of generators, and then the higher dimensional faces are simply the cliques in the Cayley graph.

► **Theorem 2.7** (Lubotzky, Samuels, Vishne [34, Theorem 1.1]). *Let  $q$  be a prime power,  $d \geq 2$ . For every  $\epsilon > 1$  the group  $G = PGL_d(\mathbb{F}_{p^\epsilon})$  has an (explicit) set of  $\binom{d}{1}_q + \binom{d}{2}_q + \dots + \binom{d}{d-1}_q$  generators, such that the Cayley complex of  $G$  with respect to these generators is a Ramanujan complex  $X$  covered by  $\mathcal{B}^{(d)}(F)$  for  $F = \mathbb{F}_q((y))$ .*

The precise definition of “Ramanujan complex” is not important for this context. For us, there are three important aspects of this theorem: efficient construction, local structure, and global structure.

**Efficient construction** Firstly, the fact that the complex is constructible in polynomial time.

**Local structure** Next, we highlight the fact that locally the complex looks like the building.

The fact that  $X$  is covered by  $\mathcal{B}^{(d)}$  means that the neighborhood of a vertex in  $X$  and in  $\mathcal{B}^{(d)}$  look exactly the same. It turns out that for the LSV complexes this continues to be true also for balls of larger radius around any vertex. This is a higher-dimensional analog of the graph property of containing no short cycles (locally looking like a tree). Define the injectivity radius of  $X$  to be the largest  $r$  such that the covering map  $\mathcal{B}^{(d)} \xrightarrow{\psi} X$  is injective from balls of radius  $\leq r$  in  $\mathcal{B}^{(d)}$  and the ball of radius  $\leq r$  in  $X$ . We do not mention the centers of the balls as they are all isomorphic.

► **Theorem 2.8** (Lubotzky and Meshulam [33], see also [15, Corollary 5.2]). *Let  $X$  be the LSV complex above. Then the injectivity radius <sup>5</sup>  $r(X)$  of  $X$  satisfies*

$$r(X) \geq \frac{\log_q |X|}{2(d-1)(d^2-1)} - \frac{1}{2}$$

where  $|X|$  is the number of vertices in  $X$ .

**Global structure** Finally, we look at the second cohomology group of the LSV complexes.

Kaufman, Kazhdan and Lubotzky [27] showed that the groups defining the LSV quotient complexes can be chosen so that the second homology is non-empty.

► **Proposition 2.9** (Kaufman, Kazhdan, Lubotzky [27, Proposition 3.6]). *There is an infinite and explicit sequence of LSV complexes with a non-vanishing second cohomology.*

We remark that Kaufman, Kazhdan and Lubotzky [27] proved that these complexes exist. To show that they are also efficiently constructible, we look into their proof to recall the construction: start with any LSV complex  $X$  viewed as a Cayley graph of a group  $G$ . Find some element of order 2 in  $G$  (such an element always exists), and then quotient  $X$  by this element, thus obtaining a complex  $Y$  that is itself is a Ramanujan complex because it is a quotient of one.  $Y$  is clearly efficiently constructible from  $X$ , and has half as many vertices. This construction shows (see [27, Proposition 3.5]) that  $H^1(Y) \neq 0$ . Furthermore, the proof of [27, Proposition 3.6] shows that because  $G$  has “property  $T$ ” one can deduce also that  $H^2(Y) \neq 0$ .

<sup>5</sup> This theorem was proven by Lubotzky and Meshulam [33]. They stated their theorem using a slightly different definition for injectivity radius but one can prove that the two definitions coincide in this case. This was reproven by Evra, Golubev and Lubotzky [15] who use the definition of injectivity radius that is convenient for us.



Evra and Kaufman proved [16, Theorem 1] that quotients of  $\mathcal{B}^{(d)}$  (and even a more general class of complexes) are so-called “cosystolic expanders” which in particular implies the following.

► **Theorem 2.10** (Evra and Kaufman [16, Part of Theorem 1]). *Let  $\{X_n\}$  be a family of LSV complexes. There exists some constant  $\mu > 0$  that depends only on  $q$  and  $d$  but not on the size  $n$  of the complex, such that every  $f \in Z^2(X) \setminus B^2(X)$  must have weight at least  $\mu \cdot |X(2)|$ .*

### 3 Main Result

#### 3.1 Local Geometry of LSV Complexes

The infinite sequence of complexes we will be working with are the LSV complexes described in Section 2.4 above. The properties we care about are (1) that they are efficiently constructible, (2) that small balls in these complexes are isomorphic to the affine building, which satisfies certain isoperimetric inequalities because it is a CAT(0) space, and (3) that each complex has a two-dimensional cocycle with linear distance from the set of coboundaries. The second and third properties provide the tension between the local and the global structure of these complexes that we now harness for our hardness.

To construct an SDP solution, we will need to show that our instance based on the LSV complex “locally looks satisfiable”. To this end, we will first develop some local properties of the LSV complex.

Note that each  $h \in C^2$  corresponds to a set of triangles. For the following statements, we consider two triangles to be connected if they share an edge. This can be used to define connected components. Note that if  $h$  can be split into connected components  $h_1, \dots, h_s$ , then the components correspond to *disjoint* sets of triangles. Moreover, no triangle in  $h_i$  shares an edge with a triangle in  $h_j$  when  $i \neq j$ , which also implies that the boundaries  $\partial h_i$  and  $\partial h_j$  correspond to disjoint sets of edges.

We prove the following claims by mapping small connected sets in  $X(2)$  to corresponding sets in the infinite building  $\mathcal{B}$ . The first proposition shows that there can be no small non-trivial cancellations (i.e., not coming from tetrahedra).

► **Proposition 3.1.** *Let  $h_0 \in C^2$  be a connected set of triangles such that  $|h_0| < r$  and  $\partial h_0 = 0$ . Then  $h_0 \in B_2$ .*

**Proof.** Since  $|h_0| < r$ , there is a ball  $N$  of radius  $r$  that contains the support of  $h_0$ . By assumption, the covering map  $\psi: \mathcal{B} \rightarrow X$  has injectivity radius of at least  $r$ . This means that there is a radius- $r$  ball  $\hat{N} = \psi^{-1}(N)$  in  $\mathcal{B}$  that is isomorphically mapped by  $\psi$  to  $N$ . Look at  $\hat{h}_0 = \psi^{-1}(h_0) \in C^2(\hat{N})$ , the chain isomorphic to  $h_0$  in the building. Clearly  $\partial \hat{h}_0 = \psi^{-1}(\partial h_0) = 0$ , and since balls in the building have zero homologies by Lemma 2.6, we deduce that  $\hat{h}_0$  itself must be a boundary, i.e. there must be some  $\hat{g}_0 \in C^3(\hat{N})$  such that  $\partial \hat{g}_0 = \hat{h}_0$ . Moving back to  $X$ , we see that  $g_0 := \psi(\hat{g}_0) \in C^3(X)$  necessarily satisfies  $\partial g_0 = h_0$ , and so  $h_0 \in B_2$ . ◀

This proposition states that locally (i.e., within the injective radius  $r$ ),  $Z_2$  looks like  $B_2$ . We thus have a complex whose cohomology group is non-trivial, yet locally, the *homology* group “looks” trivial. Note that this is a twist on what we had claimed in the introduction, a complex whose cohomology group is non-trivial, yet locally, the *cohomology* group “looks” trivial. However, these are identical statements owing to Claim 2.4.

The next proposition shows that Gromov’s filling inequality in the infinite building  $\mathcal{B}$  can be used to yield a similar consequence for small sets in the finite complex  $X$ .

► **Proposition 3.2.** *Let  $h_0 \in C_2$  be a connected set of triangles such that  $|h_0| < r$  and  $|h_0| \leq |h_0 + h|$  for all  $h \in B_2$ . Then,  $|\partial h_0| \geq c \cdot |h_0|^{1/2}$ , where  $c > 0$  is an absolute constant.*

**Proof.** As before, the support of  $h_0$  is contained in a ball  $\mathbf{N}$  of  $X$  which is isomorphic under  $\psi$  to a ball  $\hat{\mathbf{N}}$  in  $\mathcal{B}$ . Let  $\hat{h}_0 = \psi^{-1}(h_0) \in C^2(\mathcal{B})$ , and let  $\hat{f}_0 = \partial \hat{h}_0$ . We now apply the filling theorem of Gromov, which holds in  $\hat{\mathbf{N}}$  due to Lemma 2.6, to deduce that there is some  $\hat{h}_1$  that fills  $\hat{f}_0$ , namely  $\partial \hat{h}_1 = \hat{f}_0$ , and whose size is at most  $|\hat{h}_1| = O(|\hat{f}_0|^2)$ .

Now  $\partial(\hat{h}_0 - \hat{h}_1) = \hat{f}_0 - \hat{f}_0 = 0$ . Since the ball  $\hat{\mathbf{N}}$  has zero homologies by Lemma 2.6,  $\hat{h}_0 - \hat{h}_1$  itself must be a boundary: there must be some  $\hat{g} \in C^3(\hat{\mathbf{N}})$  such that  $\partial \hat{g} = \hat{h}_0 - \hat{h}_1$ . Pushing  $\hat{g}$  and  $\hat{h}_1$  back to  $X$ , we get  $g = \psi(\hat{g})$  and  $h_1 = \psi(\hat{h}_1)$ , which satisfy  $\partial g = h_0 - h_1$ . At this point we have a *small*  $h_1$  that is close via a boundary to  $h_0$ . Finally, observe that  $f_0 = \partial h_0$  satisfies  $f_0 = \psi^{-1}(\hat{f}_0)$ . So

$$|f_0| = |\hat{f}_0| \geq c \cdot |\hat{h}_1|^{1/2} = c \cdot |h_1|^{1/2} \geq c \cdot |h_0|^{1/2},$$

where the last inequality used that  $|h_0| \leq |h_0 + (h_1 - h_0)|$ , since  $h_1 - h_0 = \partial g \in B_2$ . ◀

### 3.2 Fooling $\Omega(\sqrt{\log n})$ levels of SoS Hierarchy

Let  $X$  be a  $d$ -dimensional LSV complex, with  $|X(1)| = n$  and non-trivial second cohomology group, as per Proposition 2.9. Below, we construct an instance of 3XOR in  $n$  variables using this complex, and prove a lower bound on the integrality gap of the relaxation obtained by  $\Omega(\sqrt{\log n})$  levels of the SoS hierarchy.

#### Construction

We construct a system of equations on  $X$  by putting a variable  $x_{\{a,b\}}$  for each edge  $\{a,b\} \in X(1)$  of the complex, and an equation

$$x_{\{a,b\}} + x_{\{b,c\}} + x_{\{c,a\}} = \beta_{\{a,b,c\}}$$

for each triangle  $\{a,b,c\} \in X(2)$ , where  $\beta$  is an arbitrary element of  $Z^2 \setminus B^2$ .

Recall that  $X$  can be constructed efficiently. Given  $X$ , we can find a vector  $\beta \in Z^2 \setminus B^2$  using elementary linear algebra. Therefore the entire system can be constructed efficiently.

#### Soundness

Soundness of this system follows easily from the fact that the cosystole is large.

▷ **Claim 3.3 (Soundness).** Every assignment to the system defined above falsifies at least  $\mu$  fraction of the equations.

**Proof.** An assignment to the variables is equivalent to an  $f \in C^1$ . Every equation satisfied by  $f$  is a triangle in which  $\delta f(\{a,b,c\}) = \beta_{\{a,b,c\}}$ , and so the number of unsatisfied equations is  $\text{dist}(\delta f, \beta) = |\delta f + \beta|$ . Since  $\delta f \in B^2$  and  $\beta \in Z^2 \setminus B^2$ , also  $\delta f + \beta \in Z^2 \setminus B^2$ , and so  $|\delta f + \beta|/|X(2)| \geq \text{CoSys}^2(X) \geq \mu$ . In other words, the assignment falsifies at least a  $\mu$  fraction of the equations. ◀

The main work is to prove completeness, namely to show that the system looks locally satisfiable.

### Completeness

Our main result is that this system appears satisfiable to the Sum-of-Squares hierarchy with  $O(\sqrt{\log n})$  levels. Grigoriev [20] and Schoenebeck [36] showed that to prove such a statement it suffices to analyze the *refutation width* of the system of equations (see Lemma 2.1). If the refutation width is at least  $w$ , then  $w/2$  levels of the Sum-of-Squares hierarchy cannot refute the system.

A system of linear equations over  $\mathbb{F}_2$  can be *refuted* using a proof system known as *XOR-resolution*, in which the only inference rule is: given  $\ell_1 = b_1$  and  $\ell_2 = b_2$ , deduce  $\ell_1 + \ell_2 = b_1 + b_2 \pmod{2}$ ; here  $\ell_1, \ell_2$  are XORs of variables, and  $b_1, b_2$  are constants. A refutation has the structure of a directed acyclic graph (DAG) where each non-leaf node has two incoming edges. A *refutation* is a derivation which starts with the given linear equations, placed at the leaves of a DAG, and reaches the equation  $0 = 1$  at the root of the DAG. The *width* of a linear equation  $\ell = b$  is the number of variables appearing in  $\ell$ . The width of a refutation is the maximum width of an equation in any of the nodes of the DAG.

In the remainder of this section, we prove the following theorem, which together with Lemma 2.1 implies Theorem 1.1.

► **Theorem 3.4.** *The construction above requires width at least  $\Omega(\sqrt{r})$  to refute in XOR-resolution, where  $r = \Theta(\log n)$  is the injectivity radius of the complex.*

The proof follows classical arguments of Ben-Sasson and Wigderson [8] regarding lower bounds on resolution width, which were also used in the proof of Schoenebeck [36]. Whereas Ben-Sasson and Wigderson relied on boundary expansion, we rely on Gromov’s filling inequality (and so lose a square root).

Suppose we are given a refutation for this system, and consider the corresponding DAG. Each leaf  $\nu$  in the DAG is labeled by a triangle  $T_\nu \in X(2)$ . Define

$$h_\nu := \mathbf{1}_{T_\nu} \in C^2, \quad b_\nu := \beta_{T_\nu} \in \mathbb{F}_2.$$

For each inner node  $\nu$  in the DAG, let  $\nu_1, \nu_2$  be its two incoming nodes. Define inductively,

$$h_\nu := h_{\nu_1} + h_{\nu_2} \in C^2, \quad b_\nu := b_{\nu_1} + b_{\nu_2} \in \mathbb{F}_2.$$

► **Proposition 3.5.** *For every node  $\nu$ ,  $b_\nu = \langle \beta, h_\nu \rangle$ .*

**Proof.** This is immediate by following inductively the structure of the DAG. ◀

As in [8], we next define a complexity measure for each node of the DAG. While in [8] the complexity measure is based on the number of “leaf equations” used to derive the one at a given node, we will need to discount sets of triangles corresponding to tetrahedra, as these cannot lead to contradictions. Recall that  $B_2 = \text{im } \partial_3$  is the set of triangle chains that “come from” tetrahedra chains, which we consider as the “trivial” cycles. We define a complexity measure at each node,

$$\kappa(\nu) := \text{dist}(h_\nu, B_2) = \min_{h \in B_2} |h_\nu + h|$$

that measures the distance of  $h_\nu$  from these trivial cycles. The complexity measure  $\kappa$  satisfies the following sub-additivity property.

► **Proposition 3.6.** *If  $\nu$  is an inner node in the DAG with  $\nu_1, \nu_2$  its two incoming nodes, then*

$$\kappa(\nu) \leq \kappa(\nu_1) + \kappa(\nu_2).$$

**Proof.** Let  $h_1, h_2 \in B_2$  be such that  $\kappa(\nu_1) = |h_{\nu_1} + h_1|$  and  $\kappa(\nu_2) = |h_{\nu_2} + h_2|$ . Recall that  $h_\nu = h_{\nu_1} + h_{\nu_2}$ . Then, we have

$$\begin{aligned} \kappa(\nu_1) + \kappa(\nu_2) &= |h_{\nu_1} + h_1| + |h_{\nu_2} + h_2| \geq |h_{\nu_1} + h_{\nu_2} + h_1 + h_2| \\ &= |h_\nu + h_1 + h_2| \geq \kappa(\nu). \end{aligned} \quad \blacktriangleleft$$

We also need the fact that the complexity of a node with a contradiction must be non-zero.

► **Proposition 3.7.** *If  $\kappa(\nu) = 0$  then  $b_\nu = 0$ .*

**Proof.** If  $\kappa(\nu) = 0$  then  $h_\nu \in B_2$ . Hence  $b_\nu = \langle \beta, h_\nu \rangle = 0$  since  $\beta \in Z^2 = (B_2)^\perp$  (Claim 2.3).  $\blacktriangleleft$

Next, we consider the width of each node in the DAG. For a node  $\nu$ , let

$$f_\nu := \partial h_\nu \in C^1.$$

Thus  $f_\nu$  indicates the set of variables appearing in the left-hand side of the equation on node  $\nu$ . So the width of the system is the maximum, over all nodes  $\nu$  in the DAG, of  $|f_\nu|$ .

We can now prove Theorem 3.4 using the above complexity measure, and results from Section 3.1.

**Proof of Theorem 3.4.** Let  $\nu^*$  denote the root of the DAG. By virtue of being a refutation,  $b_{\nu^*} = 1$  while  $f_{\nu^*} = 0$ . In other words,  $\partial h_{\nu^*} = f_{\nu^*} = 0$ , which means that  $h_{\nu^*} \in Z_2$ . Since  $b_{\nu^*} = 1$ , we also have by Proposition 3.7 that  $\kappa(\nu^*) > 0$ .

Let  $h \in B_2$  be such that  $\kappa(\nu^*) = |h_{\nu^*} + h|$ , and let  $h_1, \dots, h_s$  be the disjoint connected components of  $h_{\nu^*} + h$ . We will first show that  $\kappa(\nu^*) = |h_{\nu^*} + h| \geq r$ . Assuming  $\kappa(\nu^*) < r$ , we have that

$$|h_1| + \dots + |h_s| = |h_{\nu^*} + h| < r.$$

Also, since

$$\partial h_1 + \dots + \partial h_s = \partial(h_{\nu^*} + h) = \partial h_{\nu^*} = 0,$$

we must have that  $\partial h_i = 0$  for each  $i \in [s]$ , since connected components have disjoint boundaries. Applying Proposition 3.1 to each  $h_i$ , we get that  $h_i \in B_2$  for each  $i \in [s]$ . However, this implies  $h_{\nu^*} + h \in B_2$  and hence  $\kappa(\nu^*) = 0$ , which is a contradiction.

Using sub-additivity (Proposition 3.6),  $\kappa(\nu^*) \geq r$ , and the fact that the leaves of the DAG satisfy  $\kappa(\nu) = 1$ , we get that there must be some internal node  $\nu$  for which  $r/2 \leq \kappa(\nu) < r$ . We can find such a node by starting at the root and always going to the child with higher complexity, until reaching a node  $\nu$  such that  $\kappa(\nu) < r$ . We will prove that for such a node, we must have  $|f_\nu| = \Omega(\sqrt{r})$ .

As before, let  $h \in B_2$  now be such that  $\kappa(\nu) = |h_\nu + h|$ , and let  $h_1, \dots, h_s$  be the disjoint connected components of  $h_\nu + h$ . We have that  $|h_i| \leq |h_\nu + h| < r$  for each  $i \in [s]$ . By the minimality of  $|h_\nu + h|$ , we also have that for any  $h' \in B_2$  and any  $i \in [s]$ ,

$$|h_i| + |h_\nu + h - h_i| = |h_\nu + h| \leq |h_\nu + h + h'| \leq |h_i + h'| + |h_\nu + h - h_i|.$$

Thus,  $|h_i|$  is also minimal for each  $i$ , and we can apply Proposition 3.2 to each connected component  $h_i$ , to obtain

$$\begin{aligned} |f_\nu| &= |\partial(h_\nu + h)| = |\partial h_1| + \cdots + |\partial h_s| \geq c \cdot |h_1|^{1/2} + \cdots + c \cdot |h_s|^{1/2} \\ &\geq c \cdot (|h_1| + \cdots + |h_s|)^{1/2} \\ &= c \cdot |h_\nu + h|^{1/2} \geq (c/\sqrt{2}) \cdot \sqrt{r}. \quad \blacktriangleleft \end{aligned}$$

---

## References

- 1 Peter Abramenko and Kenneth S. Brown. *Buildings, Theory and applications*, volume 248 of *Graduate Texts in Mathematics*. Springer, 2008. doi:10.1007/978-0-387-78835-7.
- 2 Vedat Levi Alev, Fernando Granha Jeronimo, Dylan Quintana, Shashank Srivastava, and Madhur Tulsiani. List decoding of direct sum codes. In *Proc. 31st Annual ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 1412–1425, 2020. doi:10.1137/1.9781611975994.85.
- 3 Vedat Levi Alev, Fernando Granha Jeronimo, and Madhur Tulsiani. Approximating constraint satisfaction problems on high-dimensional expanders. In *Proc. 60th IEEE Symp. on Foundations of Comp. Science (FOCS)*, pages 180–201, 2019. doi:10.1109/FOCS.2019.00021.
- 4 Vedat Levi Alev and Lap Chi Lau. Improved analysis of higher order random walks and applications. In *Proc. 52nd ACM Symp. on Theory of Computing (STOC)*, pages 1198–1211, 2020. doi:10.1145/3357713.3384317.
- 5 Nima Anari, Kuikui Liu, and Shayan Oveis Gharan. Spectral independence in high-dimensional expanders and applications to the hardcore model. In *Proc. 61st IEEE Symp. on Foundations of Comp. Science (FOCS)*, 2020. arXiv:2001.00303.
- 6 Nima Anari, Kuikui Liu, Shayan Oveis Gharan, and Cynthia Vintant. Log-concave polynomials II: high-dimensional walks and an FPRAS for counting bases of a matroid. In *Proc. 51st ACM Symp. on Theory of Computing (STOC)*, pages 1–12, 2019. doi:10.1145/3313276.3316385.
- 7 Boaz Barak, Prasad Raghavendra, and David Steurer. Rounding semidefinite programming hierarchies via global correlation. In *Proc. 51st IEEE Symp. on Foundations of Comp. Science (FOCS)*, pages 472–481, 2011. eccc:2011/TR11-065, doi:10.1109/FOCS.2011.95.
- 8 Eli Ben-Sasson and Avi Wigderson. Short proofs are narrow - resolution made simple. *J. ACM*, 48(2):149–169, 2001. (Preliminary version in *31st STOC*, 1999). eccc:1999/TR99-022, doi:10.1145/375827.375835.
- 9 Martin R. Bridson and André Haefliger. *Metric Spaces of Non-Positive Curvature*, volume 319 of *Grundlehren der mathematischen Wissenschaften*. Springer, 1999. doi:10.1007/978-3-662-12494-9.
- 10 Siu On Chan. Approximation resistance from pairwise-independent subgroups. *J. ACM*, 63(3):27:1–27:32, 2016. (Preliminary version in *45th STOC*, 2013). eccc:2012/TR12-110, doi:10.1145/2873054.
- 11 Zongchen Chen, Kuikui Liu, and Eric Vigoda. Optimal mixing of Glauber dynamics: Entropy factorization via high-dimensional expansion. (manuscript), 2020. arXiv:2011.02075.
- 12 Yotam Dikstein and Irit Dinur. Agreement testing theorems on layered set systems. In *Proc. 60th IEEE Symp. on Foundations of Comp. Science (FOCS)*, pages 1495–1524, 2019. eccc:2019/TR19-112, doi:10.1109/FOCS.2019.00088.
- 13 Irit Dinur, Prahladh Harsha, Tali Kaufman, Inbal Livni Navon, and Amnon TaShma. List decoding with double samplers. In *Proc. 30th Annual ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 2134–2153, 2019. eccc:2018/TR18-198, doi:10.1137/1.9781611975482.129.
- 14 Irit Dinur and Tali Kaufman. High dimensional expanders imply agreement expanders. In *Proc. 58th IEEE Symp. on Foundations of Comp. Science (FOCS)*, pages 974–985, 2017. eccc:2017/TR17-089, doi:10.1109/FOCS.2017.94.
- 15 Shai Evra, Konstantin Golubev, and Alexander Lubotzky. Mixing properties and the chromatic number of Ramanujan Complexes. *Int. Math. Res. Not.*, 2015(22):11520–11548, 2015. doi:10.1093/imrn/rnv022.

- 16 Shai Evra and Tali Kaufman. Bounded degree cosystolic expanders of every dimension. In *Proc. 48th ACM Symp. on Theory of Computing (STOC)*, pages 36–48, 2016. doi:10.1145/2897518.2897543.
- 17 Shai Evra, Tali Kaufman, and Gilles Zémor. Decodable quantum LDPC codes beyond the square root distance barrier using high dimensional expanders. In *Proc. 61st IEEE Symp. on Foundations of Comp. Science (FOCS)*, 2020. arXiv:2004.07935.
- 18 Noah Fleming, Pravesh Kothari, and Toniann Pitassi. Semialgebraic proofs and efficient algorithm design. *Found. Trends Theor. Comput. Sci.*, 14(1-2):1–221, 2019. eccc:2019/TR19–106, doi:10.1561/04000000086.
- 19 Roy Gotlib and Tali Kaufman. Testing odd direct sums using high dimensional expanders. In Dimitris Achlioptas and László A. Végh, editors, *Proc. 23rd International Workshop on Randomization and Computation (RANDOM)*, volume 145 of *LIPICs*, pages 50:1–50:20. Schloss Dagstuhl, 2019. eccc:2019/TR19–124, doi:10.4230/LIPICs.APPROX-RANDOM.2019.50.
- 20 Dima Grigoriev. Linear lower bound on degrees of Positivstellensatz calculus proofs for the parity. *Theoret. Comput. Sci.*, 259(1-2):613–622, 2001. doi:10.1016/S0304-3975(00)00157-2.
- 21 Mikhael Gromov. Filling Riemannian manifolds. *J. Differential Geom.*, 18(1):1–147, 1983. doi:10.4310/jdg/1214509283.
- 22 Mikhail Gromov. Singularities, expanders and topology of maps. Part 2: from combinatorics to topology via algebraic isoperimetry. *Geom. Funct. Anal.*, 20:416–526, 2010. doi:10.1007/s00039-010-0073-8.
- 23 Anna Gundert and Uli Wagner. On eigenvalues of random complexes. *Israel J. Math.*, 216(1):545–582, 2016. doi:10.1007/s11856-016-1419-1.
- 24 Venkatesan Guruswami and Ali Kemal Sinop. Lasserre hierarchy, higher eigenvalues, and approximation schemes for graph partitioning and quadratic integer programming with PSD objectives. In *Proc. 51st IEEE Symp. on Foundations of Comp. Science (FOCS)*, pages 482–491, 2011. doi:10.1109/FOCS.2011.36.
- 25 Larry Guth. Notes on Gromov’s systolic estimate. *Geom. Dedicata*, 123:113–129, 2006. doi:10.1007/s10711-006-9111-y.
- 26 Lizhen Ji. Buildings and their applications in geometry and topology. In *Differential geometry*, volume 22 of *Adv. Lect. Math. (ALM)*, pages 89–210. Int. Press, Somerville, MA, 2012. (Expanded version of Lizhen Ji. Buildings and their Applications in Geometry and Topology. *Asian J. Math.* 10 (2006), no. 1, 11–80. doi:10.4310/AJM.2006.v10.n1.a5). URL: <http://www.math.lsa.umich.edu/~lji/ji-survey-building.pdf>.
- 27 Tali Kaufman, David Kazhdan, and Alexander Lubotzky. Isoperimetric inequalities for ramanujan complexes and topological expanders. *Geom. Funct. Anal.*, 26(1):250–287, 2016. (Preliminary version in *55th FOCS*, 2014). doi:10.1007/s00039-016-0362-y.
- 28 Tali Kaufman and Alexander Lubotzky. High dimensional expanders and property testing. In Moni Naor, editor, *Proc. 5th Innovations in Theor. Comput. Sci. (ITCS)*, pages 501–506. ACM, 2014. doi:10.1145/2554797.2554842.
- 29 Tali Kaufman and David Mass. Good distance lattices from high dimensional expanders. (manuscript), 2018. arXiv:1803.02849.
- 30 Tali Kaufman and Ran J. Tessler. New cosystolic expanders from tensors imply explicit quantum LDPC codes with  $\Omega(\sqrt{n} \log^k n)$  distance. (manuscript), 2020. arXiv:2008.09495.
- 31 Pravesh K. Kothari, Ryuhei Mori, Ryan O’Donnell, and David Witmer. Sum of squares lower bounds for refuting any CSP. In *Proc. 49th ACM Symp. on Theory of Computing (STOC)*, pages 132–145, 2017. doi:10.1145/3055399.3055485.
- 32 Nathan Linial and Roy Meshulam. Homological connectivity of random 2-complexes. *Combinatorica*, 26(4):475–487, 2006. doi:10.1007/s00493-006-0027-9.
- 33 Alexander Lubotzky and Roy Meshulam. A Moore bound for simplicial complexes. *Bull. Lond. Math. Soc.*, 39:353–358, 2007. doi:10.1112/blms/bdm003.



- 34 Alexander Lubotzky, Beth Samuels, and Uzi Vishne. Explicit constructions of Ramanujan complexes of type  $\tilde{A}_d$ . *European J. Combin.*, 26(6):965–993, 2005. doi:10.1016/j.ejc.2004.06.007.
- 35 Alexander Lubotzky, Beth Samuels, and Uzi Vishne. Ramanujan complexes of type  $\tilde{A}_d$ . *Israel J. Math.*, 149(1):267–299, 2005. doi:10.1007/BF02772543.
- 36 Grant Schoenebeck. Linear level Lasserre lower bounds for certain k-CSPs. In *Proc. 49th IEEE Symp. on Foundations of Comp. Science (FOCS)*, pages 593–602, 2008. (Full version available at <http://schoeneb.people.si.umich.edu/papers/LasserreNew.pdf>). doi:10.1109/FOCS.2008.74.
- 37 Madhur Tulsiani. CSP gaps and reductions in the Lasserre hierarchy. In *Proc. 41st ACM Symp. on Theory of Computing (STOC)*, pages 303–312, 2009. eccc:2008/TR08–104, doi:10.1145/1536414.1536457.
- 38 Stefan Wenger. A short proof of Gromov’s filling inequality. *Proc. Amer. Math. Soc.*, 136(8):2937–2941, 2008. doi:10.1090/S0002-9939-08-09203-4.

## A Proof of Lemma 2.1

► **Lemma 2.1** ([36, Lemma 13], [37, Theorem 4.2]). *Let  $\Lambda$  be a system of equations in  $n$  variables over  $\mathbb{F}_2$ , which does not admit any refutations of width at most  $2t$ . Then there exist vectors  $\{\mathbf{u}_S\}_{S \subseteq [n], |S| \leq t}$  satisfying the constraints in Figure 1, such that for all equations  $\sum_{i \in T} x_i = b_T$  in  $\Lambda$  with  $|T| \leq t$ , we have  $\langle \mathbf{u}_T, \mathbf{u}_\emptyset \rangle = (-1)^{b_T}$ .*

**Proof.** We assume that  $\Lambda$  is closed under width- $2t$  XOR-resolution, replacing  $\Lambda$  by its closure if necessary, and also that it contains the trivial equation  $0 = 0$ . We will now construct the unit vectors  $\mathbf{u}_S$ .

Define a relation  $\sim$  on subsets of  $[n]$  of size at most  $t$  as follows:  $S \sim T$  iff there exists an equation  $\sum_{i \in S \Delta T} x_i = b$  in  $\Lambda$  for some  $b \in \mathbb{F}_2$ . It is easy to check that the relation is reflexive and symmetric. It is also transitive since for  $S_1 \sim S_2$ ,  $S_2 \sim S_3$ , we can add the corresponding equations to obtain one of the form  $\sum_{i \in S_1 \Delta S_3} x_i = b$  for some  $b \in \mathbb{F}_2$ . Since  $|S_1|, |S_3| \leq t$ , this equation has at most  $2t$  variables and must be in  $\Lambda$  by the closure property. Thus, we have an equivalence relation which partitions all sets of size at most  $t$  into equivalence classes, say  $\mathcal{C}_1, \dots, \mathcal{C}_s$ . Choose an arbitrary representative  $R_i$  for each class  $\mathcal{C}_i$ , and let  $R(S)$  denote the representative for the class containing  $S$ . For convenience, we choose  $R(\emptyset) = \emptyset$ .

We now construct the SDP vectors. Let  $e_1, \dots, e_s$  be an arbitrary orthonormal set of vectors, and assign  $\mathbf{u}_{R_i} = e_i$  for all  $i \in [s]$ . Note that for any  $S$  with  $|S| \leq t$ , there must be a *unique* equation of the form  $\sum_{i \in S \Delta R(S)} x_i = b_S$  in  $\Lambda$ , since two different equations can be used to obtain a width- $2t$  refutation. We assign the vector for  $S$  as

$$\mathbf{u}_S := (-1)^{b_S} \cdot \mathbf{u}_{R(S)}.$$

The vectors are unit-length by construction. Note that if  $S_1 \Delta S_2 = S_3 \Delta S_4$ , we must have  $S_1 \sim S_2 \Leftrightarrow S_3 \sim S_4$ . If  $S_1 \not\sim S_2$ , then we have that  $\langle \mathbf{u}_{S_1}, \mathbf{u}_{S_2} \rangle = \langle \mathbf{u}_{S_3}, \mathbf{u}_{S_4} \rangle = 0$ . Otherwise, we have  $R(S_1) = R(S_2)$ ,  $R(S_3) = R(S_4)$ , and equations of the form

$$\sum_{i \in S_j \Delta R(S_j)} x_i = b_{S_j}, \quad j \in \{1, 2, 3, 4\}.$$

We must also have  $b_{S_1} + b_{S_2} = b_{S_3} + b_{S_4}$ , since otherwise we obtain two different equations with variables in  $S_1 \Delta S_2 = S_3 \Delta S_4$ , yielding a refutation. This suffices to satisfy the SDP constraints, since

$$\langle \mathbf{u}_{S_1}, \mathbf{u}_{S_2} \rangle = (-1)^{b_{S_1} + b_{S_2}} \cdot \langle \mathbf{u}_{R(S_1)}, \mathbf{u}_{R(S_2)} \rangle = (-1)^{b_{S_1} + b_{S_2}} = (-1)^{b_{S_3} + b_{S_4}} = \langle \mathbf{u}_{S_3}, \mathbf{u}_{S_4} \rangle.$$

Finally, for any equation  $\sum_{i \in T} x_i = b_T$  in  $\Lambda$  with  $|T| \leq t$ , we get  $\langle \mathbf{u}_T, \mathbf{u}_\emptyset \rangle = (-1)^{b_T}$ , since we must have  $T \sim \emptyset$  and  $R(T) = \emptyset$ . ◀



# Lower Bounds on the Running Time of Two-Way Quantum Finite Automata and Sublogarithmic-Space Quantum Turing Machines

Zachary Remscrim

Department of Computer Science, The University of Chicago, IL, USA  
remscrim@uchicago.edu

---

## Abstract

The two-way finite automaton with quantum and classical states (2QCFA), defined by Ambainis and Watrous, is a model of quantum computation whose quantum part is extremely limited; however, as they showed, 2QCFA are surprisingly powerful: a 2QCFA with only a single-qubit can recognize the language  $L_{pal} = \{w \in \{a, b\}^* : w \text{ is a palindrome}\}$  with bounded error in expected time  $2^{O(n)}$ .

We prove that their result cannot be improved upon: a 2QCFA (of any size) cannot recognize  $L_{pal}$  with bounded error in expected time  $2^{o(n)}$ . This is the first example of a language that can be recognized with bounded error by a 2QCFA in exponential time but not in subexponential time. Moreover, we prove that a quantum Turing machine (QTM) running in space  $o(\log n)$  and expected time  $2^{n^{1-\Omega(1)}}$  cannot recognize  $L_{pal}$  with bounded error; again, this is the first lower bound of its kind. Far more generally, we establish a lower bound on the running time of any 2QCFA or  $o(\log n)$ -space QTM that recognizes any language  $L$  in terms of a natural “hardness measure” of  $L$ . This allows us to exhibit a large family of languages for which we have asymptotically matching lower and upper bounds on the running time of any such 2QCFA or QTM recognizer.

**2012 ACM Subject Classification** Theory of computation → Formal languages and automata theory; Theory of computation → Quantum computation theory

**Keywords and phrases** Quantum computation, Lower bounds, Finite automata

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.39

**Related Version** Full version of the paper <https://arxiv.org/abs/2003.09877>.

**Acknowledgements** The author would like to express his sincere gratitude to Professor Michael Sipser for many years of mentorship and support, without which this work would not have been possible, and to thank Professor Richard Lipton, as well as the anonymous reviewers, for several helpful comments on an earlier draft of this paper.

## 1 Introduction

Quantum algorithms, such as Shor’s quantum polynomial time integer factorization algorithm [31], Grover’s algorithm for unstructured search [16], and the linear system solver of Harrow, Hassidim, and Lloyd [17], provide examples of natural problems on which quantum computers seem to have an advantage over their classical counterparts. However, these algorithms are designed to be run on a quantum computer that has the full power of a quantum Turing machine, whereas current experimental quantum computers only possess a rather limited quantum part. In particular, current state-of-the-art quantum computers have a very small amount of quantum memory. For example, Google’s “Sycamore” quantum computer, used in their famous recent quantum supremacy experiment [5], operates on only 53 qubits.

In this paper, we study the power quantum computers that have only a small amount of memory. We begin by considering two-way finite automata with quantum and classical states (2QCFA), originally defined by Ambainis and Watrous [2]. Informally, a 2QCFA is a two-way deterministic finite automaton (2DFA) that has been augmented by a quantum register of



© Zachary Remscrim;

licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 39; pp. 39:1–39:20

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

constant size. 2QCFA are surprisingly powerful, as originally demonstrated by Ambainis and Watrous, who showed that a 2QCFA, with only a single-qubit quantum register, can recognize, with bounded error, the language  $L_{eq} = \{a^m b^m : m \in \mathbb{N}\}$  in expected time  $O(n^4)$  and the language  $L_{pal} = \{w \in \{a, b\}^* : w \text{ is a palindrome}\}$  in expected time  $2^{O(n)}$ . In a recent paper [27], we presented further evidence of the power of few qubits by showing that 2QCFA are capable of recognizing many group word problems with bounded error.

It is known that 2QCFA are more powerful than 2DFA and two-way probabilistic finite automata (2PFA). A 2DFA can only recognize regular languages [25]. A 2PFA can recognize some nonregular languages with bounded error, given sufficient running time: in particular, a 2PFA can recognize  $L_{eq}$  with bounded error in expected time  $2^{O(n)}$  [13]. However, a 2PFA cannot recognize  $L_{eq}$  with bounded error in expected time  $2^{o(n)}$ , by a result of Greenberg and Weiss [14]; moreover, a 2PFA cannot recognize  $L_{pal}$  with bounded error in any time bound [11]. More generally, the landmark result of Dwork and Stockmeyer [10] showed that a 2PFA cannot recognize any nonregular language in expected time  $2^{n^{o(1)}}$ . In order to prove this statement, they defined a particular “hardness measure”  $D_L : \mathbb{N} \rightarrow \mathbb{N}$  of a language  $L$ . They showed that, if a 2PFA recognizes some language  $L$  with bounded error in expected time at most  $T(n)$  on all inputs of length at most  $n$ , then there is a positive real number  $a$  (that depends only on the number of states of the 2PFA), such that  $T(n) = \Omega(2^{D_L(n)^a})$  [10, Lemma 4.3]; we will refer to this statement as the “Dwork-Stockmeyer lemma.”

Very little was known about the limitations of 2QCFA. Are there any languages that a single-qubit 2QCFA can recognize with bounded error in expected exponential time but not in expected subexponential time? In particular, is it possible for a single-qubit 2QCFA to recognize  $L_{pal}$  in subexponential time, or perhaps even in polynomial time? More generally, are there any languages that a 2QCFA (that is allowed to have a quantum register of any constant size) can recognize with bounded error in exponential time but not in subexponential time? These natural questions, to our knowledge, were all open (see, for instance, [2, 3, 39] for previous discussions of these questions).

In this paper, we answer these and other related questions. We first prove an analogue of the Dwork-Stockmeyer lemma for 2QCFA.

► **Theorem 1.** *If a 2QCFA recognizes some language  $L$  with bounded error in expected time at most  $T(n)$  on all inputs of length at most  $n$ , then there a positive real number  $a$  (that depends only on the number of states of the 2QCFA), such that  $T(n) = \Omega(D_L(n)^a)$ .*

This immediately implies that the result of Ambainis and Watrous [2] cannot be improved.

► **Corollary 2.** *2QCFA (of any size) cannot recognize  $L_{pal}$  with bounded error in time  $2^{o(n)}$ .*

One of the key tools used in our proof is a quantum version of Hennie’s [18] notion of a crossing sequence, which may be of independent interest. Crossing sequences played an important role in the aforementioned 2PFA results of Dwork and Stockmeyer [10] and of Greenberg and Weiss [14]. We note that, while our lower bound on the running time of a 2QCFA is exponentially weaker than the lower bound on the running time of a 2PFA provided by the Dwork-Stockmeyer lemma, both lower bounds are in fact (asymptotically) tight; the exponential difference provides yet another example of a situation in which quantum computers have an exponential advantage over their classical counterparts. We also establish a lower bound on the expected running time of a 2QCFA recognizer of  $L$  in terms of the one-way deterministic communication complexity of testing membership in  $L$ .

We then generalize our results to prove a lower bound on the expected running time  $T(n)$  of a quantum Turing machine (QTM) that uses sublogarithmic space (i.e.,  $o(\log n)$  space) and recognizes a language  $L$  with bounded error, where this lower bound is also in terms

of  $D_L(n)$ . In particular, we show that  $L_{pal}$  cannot be recognized with bounded error by a QTM that uses sublogarithmic space and runs in expected time  $2^{n^{1-\Omega(1)}}$ . This result is particularly intriguing, as  $L_{pal}$  can be recognized by a *deterministic* TM in  $O(\log n)$  space (and, trivially, polynomial time); therefore,  $L_{pal}$  provides an example of a natural problem for which polynomial time *quantum* TMs have no (asymptotic) advantage over polynomial time *deterministic* TMs in terms of the needed amount of space.

Furthermore, we show that the class of languages recognizable with bounded error by a 2QCFA in expected polynomial time is contained in  $L/poly$ . This result, which shows that the class of languages recognizable by a particular quantum model is contained in the class of languages recognizable by a particular classical model, is a type of *dequantization* result. It is (qualitatively) similar to the Adleman-type [1] *derandomization* result  $BPL \subseteq L/poly$ , where  $BPL$  denotes the class of languages recognizable with bounded error by a probabilistic Turing machine (PTM) that uses  $O(\log n)$  space and runs in expected polynomial time. The only previous dequantization result was of a very different type: the class of languages recognizable by a 2QCFA, or more generally a QTM that uses  $O(\log n)$  space, with algebraic number transition amplitudes (even with unbounded error and with no time bound), is contained in  $DSPACE(O(\log^2 n))$  [35]. This dequantization result is analogous to the derandomization result: the class of languages recognizable by a PTM that uses  $O(\log n)$  space (even with unbounded error and with no time bound), is contained in  $DSPACE(O(\log^2 n))$  [7].

We also investigate which group word problems can be recognized by 2QCFA or QTMs with particular resource bounds. Informally, the word problem of a finitely generated group is the problem of determining if the product of a sequence of elements of that group is equal to the identity element. There is a deep connection between the algebraic properties of a finitely generated group  $G$  and the complexity of its word problem  $W_G$ , as has been demonstrated by many famous results; for example,  $W_G \in REG \Leftrightarrow G$  is finite [4],  $W_G \in CFL \Leftrightarrow G$  is virtually free [23, 9],  $W_G \in NP \Leftrightarrow G$  is a subgroup of a finitely presented group with polynomial Dehn function [6]. We have recently shown that if  $G$  is virtually abelian, then  $W_G$  may be recognized with bounded error by a single-qubit 2QCFA in polynomial time, and that, for any group  $G$  in a certain broad class of groups of exponential growth,  $W_G$  may be recognized with bounded error by a 2QCFA in time  $2^{O(n)}$  [27].

We now show that, if  $G$  has exponential growth, then  $W_G$  cannot be recognized by a 2QCFA with bounded error in time  $2^{o(n)}$ , thereby providing a broad and natural class of languages that may be recognized by a 2QCFA in time  $2^{O(n)}$  but not  $2^{o(n)}$ . We also show that, if  $W_G$  is recognizable by a 2QCFA with bounded error in expected polynomial time, then  $G$  must be virtually nilpotent (i.e.,  $G$  must have polynomial growth), thereby obtaining progress towards an exact classification of those word problems recognizable by a 2QCFA in polynomial time. Furthermore, we show analogous results for sublogarithmic-space QTMs.

## 2 Preliminaries

### 2.1 Quantum Computation

In this section, we briefly recall the fundamentals of quantum computation needed in this paper (see, for instance, [37, 24] for a more detailed presentation of the material in this section). We begin by establishing some notation. Let  $V$  denote a finite-dimensional complex Hilbert space with inner product  $\langle \cdot, \cdot \rangle : V \times V \rightarrow \mathbb{C}$ . We use the standard Dirac bra-ket notation throughout this paper. We denote elements of  $V$  by *kets*:  $|\psi\rangle, |\varphi\rangle, |q\rangle$ , etc. For the *ket*  $|\psi\rangle \in V$ , we define the corresponding *bra*  $\langle\psi| \in V^*$  to be the linear functional on  $V$  given by  $\langle\psi| \cdot \rangle : V \rightarrow \mathbb{C}$ . We write  $\langle\psi|\varphi\rangle$  to denote  $\langle\psi| \cdot \rangle(|\varphi\rangle)$ . Let  $L(V)$  denote the  $\mathbb{C}$ -vector

space consisting of all  $\mathbb{C}$ -linear maps of the form  $A : V \rightarrow V$ . For  $|\psi\rangle, |\varphi\rangle \in V$ , we define  $|\psi\rangle\langle\varphi| \in L(V)$  in the natural way: for  $|\rho\rangle \in V$ ,  $|\psi\rangle\langle\varphi|(|\rho\rangle) = |\psi\rangle\langle\varphi|\rho\rangle = \langle\varphi|\rho\rangle|\psi\rangle$ . Let  $\mathbb{1}_V \in L(V)$  denote the identity operator on  $V$  and let  $0_V \in L(V)$  denote the zero operator on  $V$ . For  $A \in L(V)$ , we define  $A^\dagger \in L(V)$ , the *Hermitian transpose* of  $A$ , to be the unique element of  $L(V)$  such that  $\langle A|\psi_1\rangle, |\psi_2\rangle\rangle = \langle|\psi_1\rangle, A^\dagger|\psi_2\rangle\rangle$ ,  $\forall |\psi_1\rangle, |\psi_2\rangle \in V$ . Let  $\text{Herm}(V) = \{A \in L(V) : A = A^\dagger\}$ ,  $\text{Pos}(V) = \{A^\dagger A : A \in L(V)\}$ ,  $\text{Proj}(V) = \{A \in \text{Pos}(V) : A^2 = A\}$ ,  $\text{U}(V) = \{A \in L(V) : AA^\dagger = \mathbb{1}_V\}$ , and  $\text{Den}(V) = \{A \in \text{Pos}(V) : \text{Tr}(A) = 1\}$  denote, respectively, the set of Hermitian, positive semi-definite, projection, unitary, and density operators on  $V$ .

A *quantum register* is specified by a finite set of *quantum basis states*  $Q = \{q_0, \dots, q_{k-1}\}$ . Corresponding to these  $k$  quantum basis states is an orthonormal basis  $\{|q_0\rangle, \dots, |q_{k-1}\rangle\}$  of the finite-dimensional complex Hilbert space  $\mathbb{C}^Q \cong \mathbb{C}^k$ . The quantum register stores a *superposition*  $|\psi\rangle = \sum_q \alpha_q |q\rangle \in \mathbb{C}^Q$ , where each  $\alpha_q \in \mathbb{C}$  and  $\sum_q |\alpha_q|^2 = 1$ ; in other words, a superposition  $|\psi\rangle$  is simply an element of  $\mathbb{C}^Q$  of norm 1.

Following the original definition of Ambainis and Watrous [2], a 2QCFA may only interact with its quantum register in two ways: by applying a *unitary transformation* or performing a *quantum measurement*. If the quantum register is currently in the superposition  $|\psi\rangle \in \mathbb{C}^Q$ , then after applying the unitary transformation  $T \in \text{U}(\mathbb{C}^Q)$ , the quantum register will be in the superposition  $T|\psi\rangle$ . A *von Neumann measurement* is specified by some  $P_1, \dots, P_l \in \text{Proj}(\mathbb{C}^Q)$ , such that  $P_i P_j = 0_{\mathbb{C}^Q}$ ,  $\forall i, j$  with  $i \neq j$ , and  $\sum_j P_j = \mathbb{1}_{\mathbb{C}^Q}$ . Quantum measurement is a probabilistic process where, if the quantum register is in the superposition  $|\psi\rangle$ , then the *result* of the measurement has the value  $r \in \{1, \dots, l\}$  with probability  $\|P_r |\psi\rangle\|^2$ ; if the result is  $r$ , then the quantum register collapses to the superposition  $\frac{1}{\|P_r |\psi\rangle\|} P_r |\psi\rangle$ . We emphasize that quantum measurement changes the state of the quantum register.

An *ensemble of pure states* of the quantum register is a set  $\{(p_i, |\psi_i\rangle) : i \in I\}$ , for some index set  $I$ , where  $p_i \in [0, 1]$  denotes the probability of the quantum register being in the superposition  $|\psi_i\rangle$ , and  $\sum_i p_i = 1$ . This ensemble corresponds to the density operator  $A = \sum_i p_i |\psi_i\rangle\langle\psi_i| \in \text{Den}(\mathbb{C}^Q)$ . Of course, many distinct ensembles correspond to the density operator  $A$ ; however, all ensembles that correspond to a particular density operator will behave the same, for our purposes (see, for instance, [24, Section 2.4] for a detailed discussion of this phenomenon, and of the following claims). That is to say, for any ensemble described by a density operator  $A \in \text{Den}(\mathbb{C}^Q)$ , applying the transformation  $T \in \text{U}(\mathbb{C}^Q)$  produces an ensemble described by the density operator  $TAT^\dagger$ . Similarly, when performing the von Neumann measurement specified by some  $P_1, \dots, P_l \in \text{Proj}(\mathbb{C}^Q)$ , the probability that the result of this measurement is  $r$  is given by  $\text{Tr}(P_r A P_r^\dagger)$ , and if the result is  $r$  then the ensemble collapses to an ensemble described by the density operator  $\frac{1}{\text{Tr}(P_r A P_r^\dagger)} P_r A P_r^\dagger$ .

Let  $V$  and  $V'$  denote a pair of finite-dimensional complex Hilbert spaces. Let  $\text{T}(V, V')$  denote the  $\mathbb{C}$ -vector space consisting of all  $\mathbb{C}$ -linear maps of the form  $\Phi : L(V) \rightarrow L(V')$ . Define  $\text{T}(V) = \text{T}(V, V)$  and let  $\mathbb{1}_{L(V)} \in \text{T}(V)$  denote the identity operator. Consider some  $\Phi \in \text{T}(V, V')$ . We say that  $\Phi$  is *positive* if,  $\forall A \in \text{Pos}(V)$ , we have  $\Phi(A) \in \text{Pos}(V')$ . We say that  $\Phi$  is *completely-positive* if, for every finite-dimensional complex Hilbert space  $W$ ,  $\Phi \otimes \mathbb{1}_{L(W)}$  is positive, where  $\otimes$  denotes the tensor product. We say that  $\Phi$  is *trace-preserving* if,  $\forall A \in L(V)$ , we have  $\text{Tr}(\Phi(A)) = \text{Tr}(A)$ . If  $\Phi$  is both completely-positive and trace-preserving, then we say  $\Phi$  is a *quantum channel*. Let  $\text{Chan}(V, V') = \{\Phi \in \text{T}(V, V') : \Phi \text{ is a quantum channel}\}$  denote the set of all such channels, and define  $\text{Chan}(V) = \text{Chan}(V, V)$ .

As we wish for our lower bound to be as strong as possible, we wish to consider a variant of the 2QCFA model that is as strong as possible; in particular, we will allow a 2QCFA to perform any physically realizable quantum operation on its quantum register.

Following Watrous [35], a *selective quantum operation*  $\mathcal{E}$  is specified by a set of operators  $\{E_{r,j} : r \in R, j \in \{1, \dots, l\}\} \subseteq L(\mathbb{C}^Q)$ , where  $R$  is a finite set and  $l \in \mathbb{N}_{\geq 1}$  (throughout the paper, we write  $\mathbb{N}_{\geq 1}$  to denote the positive natural numbers,  $\mathbb{R}_{\geq 0}$  to denote the nonnegative real numbers, etc.), such that  $\sum_{r,j} E_{r,j}^\dagger E_{r,j} = \mathbb{1}_{\mathbb{C}^Q}$ . For  $r \in R$ , we define  $\Phi_r \in T(\mathbb{C}^Q)$  such that,  $\Phi_r(A) = \sum_j E_{r,j} A E_{r,j}^\dagger, \forall A \in L(V)$ . Then, if the quantum register is described by some density operator  $A \in \text{Den}(\mathbb{C}^Q)$ , applying  $\mathcal{E}$  will have result  $r \in R$  with probability  $\text{Tr}(\Phi_r(A))$ ; if the result is  $r$ , then the quantum register is described by density operator  $\frac{1}{\text{Tr}(\Phi_r(A))} \Phi_r(A)$ . Both unitary transformations and von Neumann measurements are special cases of selective quantum operations. For any  $\mathcal{E}$ , one may always obtain a family of operators that represent  $\mathcal{E}$  with  $l \leq |Q|^2$  [37, Theorem 2.22], and therefore with  $l = |Q|^2$  (by defining any extraneous operators to be  $\mathbb{0}_{\mathbb{C}^Q}$ ). Let  $\text{QuantOp}(\mathbb{C}^Q, R)$  denote the set of all selective quantum operations specified by some  $\{E_{r,j} : r \in R, j \in \{1, \dots, |Q|^2\}\} \subseteq L(\mathbb{C}^Q)$ .

## 2.2 Definition of the 2QCFA Model

Next, we define two-way finite automata with quantum and classical states (2QCFA), essentially following the original definition of Ambainis and Watrous [2], with a few alterations that (potentially) make the model stronger. We wish to define the 2QCFA model to be as strong as possible so that our lower bounds against this model are as general as possible.

Informally, a 2QCFA is a two-way DFA that has been augmented with a quantum register of constant size; the machine may apply unitary transformations to the quantum register and perform (perhaps many) measurements of its quantum register during its computation. Formally, a 2QCFA is a 10-tuple,  $N = (Q, C, \Sigma, R, \theta, \delta, q_{\text{start}}, c_{\text{start}}, c_{\text{acc}}, c_{\text{rej}})$ , where  $Q$  is a finite set of quantum basis states,  $C$  is a finite set of classical states,  $\Sigma$  is a finite input alphabet,  $R$  is a finite set that specifies the possible results of selective quantum operations,  $\theta$  and  $\delta$  are the quantum and classical parts of the transition function,  $q_{\text{start}} \in Q$  is the quantum start state,  $c_{\text{start}} \in C$  is the classical start state, and  $c_{\text{acc}}, c_{\text{rej}} \in C$ , with  $c_{\text{acc}} \neq c_{\text{rej}}$ , specify the classical accept and reject states, respectively. We define  $\#_L, \#_R \notin \Sigma$ , with  $\#_L \neq \#_R$ , to be special symbols that serve as a left and right end-marker, respectively; we then define the tape alphabet  $\Sigma_+ = \Sigma \cup \{\#_L, \#_R\}$ . Let  $\hat{C} = C \setminus \{c_{\text{acc}}, c_{\text{rej}}\}$  denote the non-halting classical states. The components of the transition function are as follows:  $\theta : \hat{C} \times \Sigma_+ \rightarrow \text{QuantOp}(\mathbb{C}^Q, R)$  specifies the selective quantum operation that is to be performed on the quantum register and  $\delta : \hat{C} \times \Sigma_+ \times R \rightarrow C \times \{-1, 0, 1\}$  specifies how the classical state and (classical) head position evolve.

On an input  $w = w_1 \dots w_n \in \Sigma^*$ , with each  $w_i \in \Sigma$ , the 2QCFA  $N$  operates as follows. The machine has a read-only tape that contains the string  $\#_L w_1 \dots w_n \#_R$ . Initially, the classic state of  $N$  is  $c_{\text{start}}$ , the quantum register is in the superposition  $|q_{\text{start}}\rangle$ , and the head is at the left end of the tape, over the left end-marker  $\#_L$ . On each step of the computation, if the classic state is currently  $c \in \hat{C}$  and the head is over the symbol  $\sigma \in \Sigma_+$ ,  $N$  behaves as follows. First, the selective quantum operation  $\theta(c, \sigma)$  is performed on the quantum register producing some result  $r \in R$ . If the result was  $r$ , and  $\delta(c, \sigma, r) = (c', d)$ , where  $c' \in C$  and  $d \in \{-1, 0, 1\}$ , then the classical state becomes  $c'$  and the head moves left (resp. stays put, moves right) if  $d = -1$  (resp.  $d = 0, d = 1$ ).

Due to the fact that applying a selective quantum operation is a probabilistic process, the computation of  $N$  on an input  $w$  is probabilistic. We say that a 2QCFA  $N$  recognizes a language  $L$  with *two-sided bounded error*  $\epsilon$  if,  $\forall w \in L, \Pr[N \text{ accepts } w] \geq 1 - \epsilon$ , and,  $\forall w \notin L, \Pr[N \text{ accepts } w] \leq \epsilon$ . We then define  $\text{B2QCFA}(k, d, T(n), \epsilon)$  as the class of languages  $L$  for which there is a 2QCFA, with at most  $k$  quantum basis states and at most  $d$  classical states, that recognizes  $L$  with two-sided bounded error  $\epsilon$ , and has expected running time at most

$T(n)$  on all inputs of length at most  $n$ . In order to make our lower bound as strong as possible, we do *not* require  $N$  to halt with probability 1 on all  $w \in \Sigma^*$  (i.e., we permit  $N$  to reject an input by looping).

### 3 2QCFA Crossing Sequences

In this section, we develop a generalization of Hennie’s [18] notion of crossing sequences to 2QCFA, in which we make use of several ideas from the 2PFA results of Dwork and Stockmeyer [10] and Greenberg and Weiss [14]. This notion will play a key role in our proof of a lower bound on the expected running time of a 2QCFA.

When a 2QCFA  $N = (Q, C, \Sigma, R, \theta, \delta, q_{\text{start}}, c_{\text{start}}, c_{\text{acc}}, c_{\text{rej}})$  is run on an input  $w = w_1 \cdots w_n \in \Sigma^*$ , where each  $w_i \in \Sigma$ , the tape consists of  $\#_L w_1 \cdots w_n \#_R$ . One may describe the configuration of a *single probabilistic branch* of  $N$  at any particular point in time by a triple  $(A, c, h)$ , where  $A \in \text{Den}(\mathbb{C}^Q)$  describes the current state of the quantum register,  $c \in C$  is the current classical state, and  $h \in \{0, \dots, n+1\}$  is the current head position. To clarify, each step of the computation of  $N$  involves applying a selective quantum operation, which is a probabilistic process that produces a particular result  $r \in R$  with a certain probability (depending on the operation that is performed and the state of the quantum register); that is to say, the 2QCFA probabilistically branches, with a child for each  $r \in R$ .

We partition the input as  $w = xy$ , in some manner to be specified later. We then imagine running  $N$  beginning in the configuration  $(A, c, |x|)$ , where  $|x|$  denotes the length of the string  $x$  (i.e., the head is initially over the rightmost symbol of  $\#_L x$ ). We wish to describe the configuration (or, more accurately, ensemble of configurations) that  $N$  will be in when it “finishes computing” on the prefix  $\#_L x$ , either by “leaving” the string  $\#_L x$  (by moving its head right when over the rightmost symbol of  $\#_L x$ ), or by accepting or rejecting its input. Of course,  $N$  may leave  $\#_L x$ , then later reenter  $\#_L x$ , then later leave  $\#_L x$  again, and so on, which will naturally lead to our notion of a crossing sequence. Note that the string  $y$  does not affect this subcomputation as it occurs entirely within the prefix  $\#_L x$ .

More generally, we consider the case in which  $N$  is run on the prefix  $\#_L x$ , where  $N$  starts in some ensemble of configurations  $\{(p_i, (A_i, c_i, |x|)) : i \in I\}$ , where the probability of being in configuration  $(A_i, c_i, |x|)$  is given by  $p_i$  (note that the head position in each configuration is over the rightmost symbol of  $\#_L x$ ); we call this ensemble a *starting ensemble*. We then wish to describe the ensemble of configurations that  $N$  will be in when it “finishes computing” on the prefix  $\#_L x$ , (essentially) as defined above; we call this ensemble a *stopping ensemble*<sup>1</sup>. Much as it was the case that an ensemble of pure states of a quantum register can be described by a density operator, we may also describe an ensemble of configurations of a 2QCFA using density operators. This will greatly simplify our definition and analysis of the crossing sequence of a 2QCFA.

#### 3.1 Describing Ensembles of Configurations of 2QCFA

The 2QCFA  $N$  possesses both a constant-sized *quantum register*, that is described by some density operator at any particular point in time, and a constant-sized *classical register*, that stores a classical state  $c \in C$ . We can naturally interpret each  $c \in C$  as an element  $|c\rangle \in \mathbb{C}^C$ , of a special type; that is to say, each classical state  $c$  corresponds to some element  $|c\rangle$  in the

<sup>1</sup> We use the terms “starting ensemble” and “stopping ensemble” to make clear the similarity to the notion of a “starting condition” and of a “stopping condition” used by Dwork and Stockmeyer [10] in their 2PFA result.



natural orthonormal basis of  $\mathbb{C}^C$  (whereas each superposition  $|\psi\rangle$  of the quantum register corresponds to an element of  $\mathbb{C}^Q$  of norm 1). One may also view  $N$  as possessing a *head register* that stores a (classical) head position  $h \in H_x = \{0, \dots, |x| + 1\}$  (when computing on the prefix  $\#_L x$ ); of course, the size of this pseudo-register grows with the input prefix  $x$ . We analogously interpret a head position  $h \in H_x$  as being the “classical” element  $|h\rangle \in \mathbb{C}^{H_x}$ . A configuration  $(A, c, h) \in \text{Den}(\mathbb{C}^Q) \times C \times H_x$  is then simply a state of the *combined register*, which consists of the quantum, classical, and head registers.

We then consider an *ensemble of configurations*  $\{(p_i, (A_i, c_i, h_i)) : i \in I\}$ , where  $p_i$  denotes the probability of being in configuration  $(A_i, c_i, h_i)$ . We represent this ensemble (non-uniquely) by the density operator  $Z = \sum_i (p_i A_i \otimes |c_i\rangle\langle c_i| \otimes |h_i\rangle\langle h_i|) \in \text{Den}(\mathbb{C}^Q \otimes \mathbb{C}^C \otimes \mathbb{C}^{H_x})$ . Let  $\hat{i}(c, h) = \{i \in I : (c_i, h_i) = (c, h)\}$  denote the indices of those configurations in classical state  $c$  and with head position  $h$ . We then define  $p : C \times H_x \rightarrow [0, 1]$  such that  $p(c, h) = \sum_{i \in \hat{i}(c, h)} p_i$  is the total probability of being in classical state  $c$  and having head position  $h$ . We define  $A : C \times H_x \rightarrow \text{Den}(\mathbb{C}^Q)$  such that, if  $p(c, h) \neq 0$ , then  $A(c, h) = \sum_{i \in \hat{i}(c, h)} \frac{p_i}{p(c, h)} A_i$  is the density operator obtained by “merging” all density operators  $A_i$  that come from configurations  $(A_i, c_i, h_i)$  with classical state  $c_i = c$  and head position  $h_i = h$ ; if  $p(c, h) = 0$ , then we define  $A(c, h)$  arbitrarily. Then  $Z = \sum_{c, h} (p(c, h) A(c, h) \otimes |c\rangle\langle c| \otimes |h\rangle\langle h|)$ . Let  $\widehat{\text{Den}}(\mathbb{C}^Q \otimes \mathbb{C}^C \otimes \mathbb{C}^{H_x})$  denote the set of all density operators given by some  $Z$  of the above form (i.e., those density operators that respect the fact that both the classical state and head position are classical).

We also consider the case in which we are only interested in the states of the quantum and classical registers, but not the head position. We then analogously describe an ensemble  $\{(p_i, (A_i, c_i)) : i \in I\}$  by  $Z = \sum_i (p_i A_i \otimes |c_i\rangle\langle c_i|) \in \text{Den}(\mathbb{C}^Q \otimes \mathbb{C}^C)$ , and we define  $\widehat{\text{Den}}(\mathbb{C}^Q \otimes \mathbb{C}^C)$  to be the set of all such density operators. In a starting ensemble, all configurations have the same head position:  $|x|$ . We define  $I_x \in \text{T}(\mathbb{C}^Q \otimes \mathbb{C}^C, \mathbb{C}^Q \otimes \mathbb{C}^C \otimes \mathbb{C}^{H_x})$  such that  $I_x(Z) = Z \otimes |x\rangle\langle x|$ . Similarly, in a stopping ensemble, all configurations either have head position  $|x| + 1$  or are accepting or rejecting configurations (in which the head position is irrelevant). Let  $\text{Tr}_{\mathbb{C}^{H_x}} = \mathbb{1}_{\mathbb{C}^Q \otimes \mathbb{C}^C} \otimes \text{Tr} \in \text{T}(\mathbb{C}^Q \otimes \mathbb{C}^C \otimes \mathbb{C}^{H_x}, \mathbb{C}^Q \otimes \mathbb{C}^C)$  denote the *partial trace with respect to  $\mathbb{C}^{H_x}$* .

### 3.2 Definition and Properties of 2QCFA Crossing Sequences

We now formally define the notion of a crossing sequence of a 2QCFA and prove certain needed properties. We begin by establishing some notation.

► **Definition 3.** Consider a 2QCFA  $N = (Q, C, \Sigma, R, \theta, \delta, q_{\text{start}}, c_{\text{start}}, c_{\text{acc}}, c_{\text{rej}})$ . For  $c \in \widehat{C} = C \setminus \{c_{\text{acc}}, c_{\text{rej}}\}$ ,  $\sigma \in \Sigma_+ = \Sigma \sqcup \{\#_L, \#_R\}$ ,  $r \in R$ , and  $j \in J = \{1, \dots, |Q|^2\}$ , we make the following definitions.

- (i) Define  $E_{c, \sigma, r, j} \in \text{L}(\mathbb{C}^Q)$  such that  $\theta(c, \sigma) \in \text{QuantOp}(\mathbb{C}^Q, R)$  is described by  $\{E_{c, \sigma, r, j} : r \in R, j \in J\}$ .
- (ii) Define  $\Phi_{c, \sigma, r} \in \text{T}(\mathbb{C}^Q)$  such that  $\Phi_{c, \sigma, r}(A) = \sum_j E_{c, \sigma, r, j} A E_{c, \sigma, r, j}^\dagger$ ,  $\forall A \in \text{L}(\mathbb{C}^Q)$ .
- (iii) Let  $\gamma_{c, \sigma, r} \in C$  and  $d_{c, \sigma, r} \in \{-1, 0, 1\}$  denote, respectively, the new classical state and the motion of the head, if the result of applying  $\theta(c, \sigma)$  is  $r$ ; i.e.,  $\delta(c, \sigma, r) = (\gamma_{c, \sigma, r}, d_{c, \sigma, r})$ .

Consider some  $x \in \Sigma^*$ . Let  $\widehat{H}_x = \{0, \dots, |x|\}$  denote the head positions corresponding to the prefix  $\#_L x$ , and let  $H_x = \{0, \dots, |x| + 1\}$  denote the set of possible positions the head of  $N$  may be in until it “finishes computing” on the prefix  $\#_L x$ . We define an operator  $S_x \in \text{T}(\mathbb{C}^Q \otimes \mathbb{C}^C \otimes \mathbb{C}^{H_x})$  that describes a single step of the computation of  $N$  on  $\#_L x$ , as follows. If  $(c, h) \in \widehat{C} \times \widehat{H}_x$ , then  $S_x(A \otimes |c\rangle\langle c| \otimes |h\rangle\langle h|)$  describes the ensemble of configurations of  $N$  after running  $N$  for a single step beginning in the configuration  $(A, c, h)$ ;



otherwise (i.e., if  $c \in \{c_{\text{acc}}, c_{\text{rej}}\}$  or  $h = |x| + 1$ , which means  $N$  has “finished computing” on  $\#_L x$ )  $S_x$  leaves the configuration unchanged. We will observe that  $S_x$  correctly describes the behavior of  $N$  on an ensemble of configurations, and that  $S_x$  is a quantum channel.

► **Definition 4.** Using the notation of Definition 3, consider a 2QCFA  $N$  and a string  $x \in \Sigma^*$ . Let  $x_h \in \Sigma$  denote the symbol of  $x$  at position  $h$ , and let  $x_0 = \#_L$  denote the left end-marker.

(i) For  $(c, h, r, j) \in C \times H_x \times R \times J$ , define  $\tilde{E}_{x,c,h,r,j} \in L(\mathbb{C}^Q \otimes \mathbb{C}^C \otimes \mathbb{C}^{H_x})$  as follows.

$$\tilde{E}_{x,c,h,r,j} = \begin{cases} E_{c,x_h,r,j} \otimes |\gamma_{c,x_h,r}\rangle \langle c| \otimes |h + d_{c,x_h,r}\rangle \langle h|, & \text{if } (c, h) \in \hat{C} \times \hat{H} \\ \frac{1}{\sqrt{|R||J|}} \mathbb{1}_{\mathbb{C}^Q} \otimes |c\rangle \langle c| \otimes |h\rangle \langle h|, & \text{otherwise.} \end{cases}$$

(ii) Define  $S_x \in T(\mathbb{C}^Q \otimes \mathbb{C}^C \otimes \mathbb{C}^{H_x})$  such that

$$S_x(Z) = \sum_{(c,h,r,j) \in C \times H_x \times R \times J} \tilde{E}_{x,c,h,r,j} Z \tilde{E}_{x,c,h,r,j}^\dagger, \quad \forall Z \in L(\mathbb{C}^Q \otimes \mathbb{C}^C \otimes \mathbb{C}^{H_x}).$$

► **Lemma 5.** Using the above notation, consider some  $x \in \Sigma^*$  and  $(A, \hat{c}, \hat{h}) \in \text{Den}(\mathbb{C}^Q) \times \hat{C} \times \hat{H}_x$ . Let  $\hat{Z} = A \otimes |\hat{c}\rangle \langle \hat{c}| \otimes |\hat{h}\rangle \langle \hat{h}|$ .  $S_x(\hat{Z})$  describes the ensemble of configurations obtained after running  $N$  for one step, beginning in the configuration  $(A, \hat{c}, \hat{h})$ , on input prefix  $\#_L x$ .

**Proof.** Let  $\tilde{R}_{x,c,\hat{h},A} = \{r \in R : \text{Tr}(\Phi_{c,x_{\hat{h}},r}(A)) \neq 0\}$ . Note that  $A \in \text{Den}(\mathbb{C}^Q) \subseteq \text{Pos}(\mathbb{C}^Q)$ , which implies  $\Phi_{c,x_{\hat{h}},r}(A) \in \text{Pos}(\mathbb{C}^Q)$ ; therefore, we have  $\text{Tr}(\Phi_{c,x_{\hat{h}},r}(A)) = 0$  precisely when  $\Phi_{c,x_{\hat{h}},r}(A) = 0_{\mathbb{C}^Q}$ . After running  $N$  as described, it is in an ensemble of configurations

$$\left\{ \left( \text{Tr}(\Phi_{c,x_{\hat{h}},r}(A)), \left( \frac{1}{\text{Tr}(\Phi_{c,x_{\hat{h}},r}(A))} \Phi_{c,x_{\hat{h}},r}(A), \gamma_{c,x_{\hat{h}},r}, \hat{h} + d_{c,x_{\hat{h}},r} \right) \right) : r \in \tilde{R}_{x,c,\hat{h},A} \right\}.$$

This ensemble of configurations is described by the density operator  $\hat{Z}'$  given by

$$\begin{aligned} \hat{Z}' &= \sum_{r \in \tilde{R}_{x,c,\hat{h},A}} \left( \frac{\text{Tr}(\Phi_{c,x_{\hat{h}},r}(A))}{\text{Tr}(\Phi_{c,x_{\hat{h}},r}(A))} \Phi_{c,x_{\hat{h}},r}(A) \otimes |\gamma_{c,x_{\hat{h}},r}\rangle \langle \gamma_{c,x_{\hat{h}},r}| \otimes |\hat{h} + d_{c,x_{\hat{h}},r}\rangle \langle \hat{h} + d_{c,x_{\hat{h}},r}| \right) \\ &= \sum_{r \in R} \left( \Phi_{c,x_{\hat{h}},r}(A) \otimes |\gamma_{c,x_{\hat{h}},r}\rangle \langle \gamma_{c,x_{\hat{h}},r}| \otimes |\hat{h} + d_{c,x_{\hat{h}},r}\rangle \langle \hat{h} + d_{c,x_{\hat{h}},r}| \right). \end{aligned}$$

Let  $B_{x,c,\hat{h},r} = |\gamma_{c,x_{\hat{h}},r}\rangle \langle \gamma_{c,x_{\hat{h}},r}| \otimes |\hat{h} + d_{c,x_{\hat{h}},r}\rangle \langle \hat{h} + d_{c,x_{\hat{h}},r}|$ . If  $(c, h) \in \hat{C} \times \hat{H}_x$ , then

$$\begin{aligned} \tilde{E}_{x,c,h,r,j} \hat{Z} \tilde{E}_{x,c,h,r,j}^\dagger &= \tilde{E}_{x,c,h,r,j} (A \otimes |\hat{c}\rangle \langle \hat{c}| \otimes |\hat{h}\rangle \langle \hat{h}|) \tilde{E}_{x,c,h,r,j}^\dagger \\ &= E_{c,x_h,r,j} A E_{c,x_h,r,j}^\dagger \otimes |\gamma_{c,x_h,r}\rangle \langle c| \langle \hat{c}| \langle \gamma_{c,x_h,r}| \otimes |h + d_{c,x_h,r}\rangle \langle \hat{h}| \langle \hat{h}| \langle h + d_{c,x_h,r}| \\ &= \begin{cases} E_{c,x_{\hat{h}},r,j} A E_{c,x_{\hat{h}},r,j}^\dagger \otimes B_{x,c,\hat{h},r}, & \text{if } (c, h) = (\hat{c}, \hat{h}) \\ 0_{\mathbb{C}^Q \otimes \mathbb{C}^C \otimes \mathbb{C}^{H_x}}, & \text{otherwise.} \end{cases} \end{aligned}$$

If, instead,  $(c, h) \notin \hat{C} \times \hat{H}_x$ , then  $\tilde{E}_{x,c,h,r,j} \hat{Z} \tilde{E}_{x,c,h,r,j}^\dagger = 0_{\mathbb{C}^Q \otimes \mathbb{C}^C \otimes \mathbb{C}^{H_x}}$ . Therefore

$$\begin{aligned} S_x(\hat{Z}) &= \sum_{(r,j) \in R \times J} \sum_{(c,h) \in C \times H_x} \tilde{E}_{x,c,h,r,j} \hat{Z} \tilde{E}_{x,c,h,r,j}^\dagger = \sum_{(r,j) \in R \times J} \left( E_{c,x_{\hat{h}},r,j} A E_{c,x_{\hat{h}},r,j}^\dagger \otimes B_{x,c,\hat{h},r} \right) \\ &= \sum_{r \in R} \left( \left( \sum_{j \in J} E_{c,x_{\hat{h}},r,j} A E_{c,x_{\hat{h}},r,j}^\dagger \right) \otimes B_{x,c,\hat{h},r} \right) = \sum_{r \in R} \left( \Phi_{c,x_{\hat{h}},r}(A) \otimes B_{x,c,\hat{h},r} \right) = \hat{Z}'. \quad \blacktriangleleft \end{aligned}$$

► **Lemma 6.** Consider some  $x \in \Sigma^*$  and  $Z \in \widehat{\text{Den}}(\mathbb{C}^Q \otimes \mathbb{C}^C \otimes \mathbb{C}^{H_x})$ . If  $\{(p_i, (A_i, c_i, h_i)) : i \in I\}$  is some ensemble of configurations described by  $Z$ , then  $S_x(Z)$  describes the ensemble of configurations obtained by replacing each configuration with  $(c_i, h_i) \in (\widehat{C} \times \widehat{H}_x)$  by the ensemble (scaled by  $p_i$ ) of configurations obtained by running  $N$  for one step beginning in the configuration  $(A_i, c_i, h_i)$ , and leaving each configuration with  $(c_i, h_i) \notin (\widehat{C} \times \widehat{H}_x)$  unchanged.

**Proof.** This follows immediately from Lemma 5 and linearity. ◀

► **Lemma 7.**  $S_x \in \text{Chan}(\mathbb{C}^Q \otimes \mathbb{C}^C \otimes \mathbb{C}^{H_x})$ ,  $\forall x \in \Sigma^*$ .

**Proof.**  $\{\tilde{E}_{x,c,h,r,j} : (c, h, r, j) \in C \times H_x \times R \times J\}$  is a Kraus representation of  $S_x$ ; therefore,  $S_x \in \text{Chan}(\mathbb{C}^Q \otimes \mathbb{C}^C \otimes \mathbb{C}^{H_x}) \Leftrightarrow \sum_{c,h,r,j} \tilde{E}_{x,c,h,r,j}^\dagger \tilde{E}_{x,c,h,r,j} = \mathbb{1}$  [37, Corollary 2.27]. This latter statement follows from a straightforward calculation; see the full paper [26] for a proof. ◀

For  $m \in \mathbb{N}$ , we define the  $m$ -truncated stopping ensemble as the ensemble of configurations that  $N$  will be in when it “finishes computing” on  $\#_L x$ , as defined earlier, with the modification that if any particular branch of  $N$  runs for more than  $m$  steps, the computation of that branch will be “interrupted” immediately before it attempts to perform the  $m+1^{\text{st}}$  step and instead immediately reject. To be clear, this truncation occurs only in the *analysis* of  $N$ ; we do not modify the 2QCFA. The following truncation operator  $T_x$ , which terminates all branches on which  $N$  has not yet “finished computing,” will help us do this.

► **Definition 8.** For  $(c, h) \in (C, H_x)$ , let  $\hat{E}_{x,c,h} = \mathbb{1}_{\mathbb{C}^Q} \otimes |c'\rangle\langle c'| \otimes |h\rangle\langle h|$ , where  $c' = c_{\text{rej}}$  if  $(c, h) \in \widehat{C} \times \widehat{H}_x$ , and  $c' = c$  otherwise. We then define  $T_x \in \text{T}(\mathbb{C}^Q \otimes \mathbb{C}^C \otimes \mathbb{C}^{H_x})$  such that  $T_x(Z) = \sum_{(c,h) \in C \times H_x} \hat{E}_{x,c,h} Z \hat{E}_{x,c,h}^\dagger$ .

► **Lemma 9.** Using the above notation, the following statements hold.

- (i) For any  $Z \in \widehat{\text{Den}}(\mathbb{C}^Q \otimes \mathbb{C}^C \otimes \mathbb{C}^{H_x})$ , if  $\{(p_i, (A_i, c_i, h_i)) : i \in I\}$  is any ensemble of configurations described by  $Z$ , then  $T_x(Z)$  describes the ensemble of configurations in which each configuration with  $(c_i, h_i) \in \widehat{C} \times \widehat{H}_x$  is replaced by the configuration  $(A_i, c_{\text{rej}}, h_i)$  (i.e., all configurations in which  $N$  has not yet “finished computing” on  $\#_L x$  become rejecting configurations) and all other configurations are left unchanged.
- (ii)  $T_x \in \text{Chan}(\mathbb{C}^Q \otimes \mathbb{C}^C \otimes \mathbb{C}^{H_x})$ .

**Proof.**

- (i) Immediate from definitions.
- (ii) As in the proof of Lemma 7, we may straightforwardly show  $\sum_{c,h} \hat{E}_{x,c,h}^\dagger \hat{E}_{x,c,h} = \mathbb{1}_{\mathbb{C}^Q \otimes \mathbb{C}^C \otimes \mathbb{C}^{H_x}}$ , which implies  $T_x \in \text{Chan}(\mathbb{C}^Q \otimes \mathbb{C}^C \otimes \mathbb{C}^{H_x})$  [37, Corollary 2.27]. ◀

The following operator converts starting ensembles to  $m$ -truncated stopping ensembles.

► **Definition 10.** For  $x \in \Sigma^*$  and  $m \in \mathbb{N}$ , we define the  $m$ -truncated transfer operator  $N_{x,m}^\Leftarrow = \text{Tr}_{\mathbb{C}^{H_x}} \circ T_x \circ S_x^m \circ I_x \in \text{T}(\mathbb{C}^Q \otimes \mathbb{C}^C)$ . For  $y \in \Sigma^*$ , we next consider the “dual case” of running  $N$  on the suffix  $y\#_R$  beginning in some ensemble of configurations  $\{(p_i, (A_i, c_i, |x|+1)) : i \in I\}$  (i.e., the head position of every configuration is over the leftmost symbol of  $y\#_R$ ). We define the notion of an  $m$ -truncated stopping ensemble, and all other notions, symmetrically. That is to say, a branch of  $N$  “finishes computing” on  $y\#_R$  when it either “leaves”  $y\#_R$  (by moving its head left from the leftmost symbol of  $y\#_R$ ), or accepts or rejects the input, or runs for more than  $m$  steps. We then define  $N_{y,m}^\Leftarrow \in \text{T}(\mathbb{C}^Q \otimes \mathbb{C}^C)$  as the corresponding “dual”  $m$ -truncated transfer operator for  $y$ .

► **Lemma 11.** *Using the notation of Definition 10, the following statements hold.*

- (i) *For  $Z \in \widehat{\text{Den}}(\mathbb{C}^Q \otimes \mathbb{C}^C)$ , if  $N$  is run on  $\#_L x$  beginning in any ensemble of configurations described by  $I_x(Z)$  (i.e., the head position of every configuration is over the rightmost symbol of  $\#_L x$ ), then the  $m$ -truncated stopping ensemble is described by  $N_{x,m}^{\leftarrow}(Z)$ .*
- (ii) *For  $Z \in \widehat{\text{Den}}(\mathbb{C}^Q \otimes \mathbb{C}^C)$ , if  $N$  is run on  $y\#_R$  beginning in any ensemble of configurations described by  $I_{x+1}(Z)$ , then the  $m$ -truncated stopping ensemble is described by  $N_{y,m}^{\rightarrow}(Z)$ .*
- (iii) *We have  $N_{x,m}^{\leftarrow}, N_{y,m}^{\rightarrow} \in \text{Chan}(\mathbb{C}^Q \otimes \mathbb{C}^C)$ ,  $\forall x, y \in \Sigma^*, \forall m \in \mathbb{N}$ .*

**Proof.**

- (i) Immediate by Definition 10, Lemma 6, and Lemma 9(i).
- (ii) Immediate by Definition 10, and analogous versions of Lemma 6, and Lemma 9(i).
- (iii) By definition,  $N_{x,m}^{\leftarrow} = \text{Tr}_{\mathbb{C}^{H_x}} \circ T_x \circ S_x^m \circ I_x$ . By Lemma 7 and Lemma 9(ii), we have  $S_x, T_x \in \text{Chan}(\mathbb{C}^Q \otimes \mathbb{C}^C \otimes \mathbb{C}^{H_x})$ . It is straightforward to see that  $I_x \in \text{Chan}(\mathbb{C}^Q \otimes \mathbb{C}^C, \mathbb{C}^Q \otimes \mathbb{C}^C \otimes \mathbb{C}^{H_x})$  and  $\text{Tr}_{\mathbb{C}^{H_x}} \in \text{Chan}(\mathbb{C}^Q \otimes \mathbb{C}^C \otimes \mathbb{C}^{H_x}, \mathbb{C}^Q \otimes \mathbb{C}^C)$  and that the composition of quantum channels is a quantum channel (see, for instance, [37, Section 2.2]). The claim for  $N_{y,m}^{\rightarrow}$  follows by an analogous argument. ◀

Given a 2QCFA  $N$ , we produce an equivalent  $N'$  of a certain convenient form, in much the same way that Dwork and Stockmeyer [10] converted a 2PFA to a convenient form. The 2QCFA  $N'$  is identical to  $N$ , except for the addition of two new classical states,  $c'_{\text{start}}$  and  $c'$ , where  $c'_{\text{start}}$  will be the start state of  $N'$ . On any input,  $N'$  will move its head to the right until it reaches  $\#_R$ , performing the trivial transformation to its quantum register along the way. When it reaches  $\#_R$ ,  $N'$  will enter  $c'$ ; then,  $N'$  will move its head to the left until it reaches  $\#_L$ , again performing the trivial transformation to its quantum register. When it reaches  $\#_L$ ,  $N'$  will enter the original start state  $c_{\text{start}}$  and behave identically to  $N$  from this point. For the remainder of the paper, we assume all 2QCFA have this form.

Finally, we define the  $m$ -truncated crossing sequence.

► **Definition 12.** For  $x, y \in \Sigma^*$  and  $m \in \mathbb{N}$ , the  $m$ -truncated crossing sequence of  $N$  with respect to the (partitioned) input  $xy$  is the sequence  $Z_1, Z_2, \dots \in \widehat{\text{Den}}(\mathbb{C}^Q \otimes \mathbb{C}^C)$ , defined as follows. The density operator  $Z_1$  describes the ensemble consisting of the single configuration (of the quantum register and classical register)  $(|q_{\text{start}}\rangle, c_{\text{start}})$  that  $N$  is in when it first crosses from  $\#_L x$  into  $y\#_R$ , which is of this simple form due to the assumed form of  $N$ . The sequence  $Z_1, Z_2, \dots$  is then obtained by starting with  $Z_1$  and alternately applying  $N_{y,m}^{\rightarrow}$  and  $N_{x,m}^{\leftarrow}$ . To be precise,

$$Z_i = \begin{cases} |q_{\text{start}}\rangle \langle q_{\text{start}}| \otimes |c_{\text{start}}\rangle \langle c_{\text{start}}|, & i = 1 \\ N_{y,m}^{\rightarrow}(Z_{i-1}), & i > 1, i \text{ is even} \\ N_{x,m}^{\leftarrow}(Z_{i-1}), & i > 1, i \text{ is odd.} \end{cases}$$

► **Remark.** Note that the  $\{Z_i\}$  that comprise a crossing sequence do *not* describe the ensemble of configurations of  $N$  at particular points in time during its computation on the input  $xy$ ; instead,  $Z_i$  describes the ensemble of configurations of the set of all the probabilistic branches of  $N$  at the  $i^{\text{th}}$  time each branch crosses between  $\#_L x$  and  $y\#_R$ .

## 4 Lower Bounds on the Running Time of 2QCFA

Dwork and Stockmeyer proved a lower bound [10, Lemma 4.3] on the expected running time  $T(n)$  of any 2PFA that recognizes any language  $L$  with bounded error, in terms of their hardness measure  $D_L(n)$ . We prove that an analogous claim holds for any 2QCFA.

The preceding quantum generalization of a crossing sequence plays a key role in the proof, essentially taking the place of the Markov chains used both in the aforementioned result of Dwork and Stockmeyer and in the earlier result of Greenberg and Weiss [14], which showed that 2PFA cannot recognize  $L_{eq}$  in subexponential time.

## 4.1 Nonregularity

For a language  $L$ , Dwork and Stockmeyer [10] defined a particular “hardness measure”  $D_L : \mathbb{N} \rightarrow \mathbb{N}$ , which they called the *nonregularity* of  $L$ , as follows. Let  $\Sigma$  be a finite alphabet,  $L \subseteq \Sigma^*$  a language, and  $n \in \mathbb{N}$ . Let  $\Sigma^{\leq n} = \{w \in \Sigma^* : |w| \leq n\}$  denote the set of all strings over  $\Sigma$  of length at most  $n$  and consider some  $x, x' \in \Sigma^{\leq n}$ . We say that  $x$  and  $x'$  are  $(L, n)$ -dissimilar, which we denote by writing  $x \not\sim_{L,n} x'$ , if  $\exists y \in \Sigma^{\leq n - \max(|x|, |x'|)}$ , such that  $xy \in L \Leftrightarrow x'y \notin L$ . Recall the classic Myhill-Nerode inequivalence relation, in which  $x, x' \in \Sigma^*$  are  $L$ -dissimilar if  $\exists y \in \Sigma^*$ , such that  $xy \in L \Leftrightarrow x'y \notin L$ . Then  $x, x' \in \Sigma^{\leq n}$  are  $(L, n)$ -dissimilar precisely when they are  $L$ -dissimilar, and the dissimilarity is witnessed by a “short” string  $y$ . We then define  $D_L(n)$  to be the largest  $h \in \mathbb{N}$  such that  $\exists x_1, \dots, x_h \in \Sigma^{\leq n}$  that are pairwise  $(L, n)$ -dissimilar (i.e.,  $x_i \not\sim_{L,n} x_j$ ,  $\forall i, j$  with  $i \neq j$ ).

In fact,  $D_L$  has been defined by many authors, both before and after Dwork and Stockmeyer, who gave many different names to this quantity and who (repeatedly) rediscovered certain basic facts about it; we refer the reader to the excellent paper of Shallit and Breitbart [30] for a detailed history of the study of  $D_L$  and related hardness measures.

## 4.2 A 2QCFA Analogue of the Dwork-Stockmeyer Lemma

We now prove that an analogue of the Dwork-Stockmeyer lemma holds for 2QCFA. The main idea is as follows. Suppose the 2QCFA  $N$  recognizes  $L \subseteq \Sigma^*$ , with two-sided bounded error  $\epsilon$ , in expected time at most  $T(n)$ . We show that, if  $D_L(n)$  is “large,” then, for any  $m \in \mathbb{N}$ , we can find  $x, x' \in \Sigma^{\leq n}$  such that  $x \not\sim_{L,n} x'$  and the distance between the corresponding  $m$ -truncated transfer operators  $N_{x,m}^{\leftarrow}$  and  $N_{x',m}^{\leftarrow}$  is “small.” By definition,  $\exists y \in \Sigma^{\leq n - \max(|x|, |x'|)}$ , such that  $xy \in L \Leftrightarrow x'y \notin L$ ; note that  $xy, x'y \in \Sigma^{\leq n}$ . Without loss of generality, we assume  $xy \in L$ , and hence  $x'y \notin L$ . We also show that, for  $m$  sufficiently large, if the distance between  $N_{x,m}^{\leftarrow}$  and  $N_{x',m}^{\leftarrow}$  is “small,” then the behavior of  $N$  on the partitioned inputs  $xy$  and  $x'y$  will be similar; in particular, if  $T(n)$  is “small,” then  $\Pr[N \text{ accepts } xy] \approx \Pr[N \text{ accepts } x'y]$ . However, as  $xy \in L$ , we must have  $\Pr[N \text{ accepts } xy] \geq 1 - \epsilon$ , and as  $x'y \notin L$ , we must have  $\Pr[N \text{ accepts } x'y] \leq \epsilon$ , which is impossible. This contradiction allows us to establish a lower bound on  $T(n)$  in terms of  $D_L(n)$ . In this section, we formalize this idea.

Recall that the *trace norm*  $\|\cdot\|_1 : L(V) \rightarrow \mathbb{R}_{\geq 0}$  is given by  $\|Z\|_1 = \text{Tr}(\sqrt{Z^\dagger Z})$ ,  $\forall Z \in L(V)$ , and the *induced trace norm*  $\|\cdot\|_1 : T(V, V') \rightarrow \mathbb{R}_{\geq 0}$ , is given  $\|\Phi\|_1 = \sup\{\|\Phi(Z)\|_1 : Z \in L(V), \|Z\|_1 \leq 1\}$ ,  $\forall \Phi \in T(V, V')$ . Suppose  $N$  is run on two distinct partitioned inputs  $xy$  and  $x'y$ , producing two distinct  $m$ -truncated crossing sequences, following Definition 12. We first show that if  $\|N_{x,m}^{\leftarrow} - N_{x',m}^{\leftarrow}\|_1$  is “small,” then these crossing sequences are similar.

► **Lemma 13.** *Consider a 2QCFA  $N$  with quantum basis states  $Q$ , classical states  $C$ , and input alphabet  $\Sigma$ . For  $x, x', y \in \Sigma^*$  and  $m \in \mathbb{N}$ , let  $Z_1, Z_2, \dots \in \widehat{\text{Den}}(\mathbb{C}^Q \otimes \mathbb{C}^C)$  (resp.  $Z'_1, Z'_2, \dots \in \widehat{\text{Den}}(\mathbb{C}^Q \otimes \mathbb{C}^C)$ ) denote the  $m$ -truncated crossing sequence obtained when  $N$  is run on  $xy$  (resp.  $x'y$ ). Then  $\|Z_i - Z'_i\|_1 \leq \lfloor \frac{i-1}{2} \rfloor \|N_{x,m}^{\leftarrow} - N_{x',m}^{\leftarrow}\|_1$ ,  $\forall i \in \mathbb{N}_{\geq 1}$ .*

**Proof.** By definition,  $Z_1 = |q_{\text{start}}\rangle \langle q_{\text{start}}| \otimes |c_{\text{start}}\rangle \langle c_{\text{start}}| = Z'_1$ , and so  $\|Z_1 - Z'_1\|_1 = 0$ . Note that  $\|\Phi(Z)\|_1 \leq \|Z\|_1$ ,  $\forall Z \in L(\mathbb{C}^Q \otimes \mathbb{C}^C)$ ,  $\forall \Phi \in \text{Chan}(\mathbb{C}^Q \otimes \mathbb{C}^C)$  [37, Corollary 3.40]. Therefore, for any  $\Phi \in \text{Chan}(\mathbb{C}^Q \otimes \mathbb{C}^C)$  and any  $Z, Z' \in L(\mathbb{C}^Q \otimes \mathbb{C}^C)$ , we have

$\|\Phi(Z) - \Phi(Z')\|_1 = \|\Phi(Z - Z')\|_1 \leq \|Z - Z'\|_1$ . By Lemma 11(iii),  $N_{x,m}^{\leftarrow}, N_{x',m}^{\leftarrow}, N_{y,m}^{\rightarrow} \in \text{Chan}(\mathbb{C}^Q \otimes \mathbb{C}^C)$ . For  $i$  even,  $Z_i = N_{y,m}^{\rightarrow}(Z_{i-1})$  and  $Z'_i = N_{y,m}^{\rightarrow}(Z'_{i-1})$ . We then have

$$\|Z_i - Z'_i\|_1 = \|N_{y,m}^{\rightarrow}(Z_{i-1}) - N_{y,m}^{\rightarrow}(Z'_{i-1})\|_1 \leq \|Z_{i-1} - Z'_{i-1}\|_1.$$

For odd  $i > 1$ ,  $Z_i = N_{x,m}^{\leftarrow}(Z_{i-1})$  and  $Z'_i = N_{x',m}^{\leftarrow}(Z'_{i-1})$ . We have  $\|Z\|_1 = 1$ ,  $\forall Z \in \text{Den}(\mathbb{C}^Q \otimes \mathbb{C}^C)$ , which implies  $\|\Phi(Z)\|_1 \leq \|\Phi\|_1$ ,  $\forall \Phi \in \text{T}(\mathbb{C}^Q \otimes \mathbb{C}^C)$ . Therefore,

$$\begin{aligned} \|Z_i - Z'_i\|_1 &= \|N_{x,m}^{\leftarrow}(Z_{i-1}) - N_{x',m}^{\leftarrow}(Z'_{i-1})\|_1 \\ &\leq \|N_{x,m}^{\leftarrow}(Z_{i-1}) - N_{x,m}^{\leftarrow}(Z'_{i-1})\|_1 + \|N_{x,m}^{\leftarrow}(Z'_{i-1}) - N_{x',m}^{\leftarrow}(Z'_{i-1})\|_1 \\ &= \|N_{x,m}^{\leftarrow}(Z_{i-1} - Z'_{i-1})\|_1 + \|(N_{x,m}^{\leftarrow} - N_{x',m}^{\leftarrow})(Z'_{i-1})\|_1 \leq \|Z_{i-1} - Z'_{i-1}\|_1 + \|N_{x,m}^{\leftarrow} - N_{x',m}^{\leftarrow}\|_1 \end{aligned}$$

The claim then follows by induction on  $i \in \mathbb{N}_{\geq 1}$ .  $\blacktriangleleft$

► **Lemma 14.** *Consider a language  $L \subseteq \Sigma^*$ . Suppose  $L \in \text{B2QCFA}(k, d, T(n), \epsilon)$ , for some  $k, d \in \mathbb{N}_{\geq 2}$ ,  $T : \mathbb{N} \rightarrow \mathbb{N}$ , and  $\epsilon \in [0, \frac{1}{2}]$ . If, for some  $n \in \mathbb{N}$ ,  $\exists x, x' \in \Sigma^{\leq n}$  such that  $x \not\sim_{L,n} x'$ , then  $T(n) \geq \frac{(1-2\epsilon)^2}{2} \|N_{x,m}^{\leftarrow} - N_{x',m}^{\leftarrow}\|_1^{-1}$ ,  $\forall m \geq \lceil \frac{2}{1-2\epsilon} T(n) \rceil$ .*

**Proof.** By definition,  $x \not\sim_{L,n} x'$  precisely when  $\exists y \in \Sigma^*$  such that  $xy, x'y \in \Sigma^{\leq n}$ , and  $xy \in L \Leftrightarrow x'y \notin L$ . Fix such a  $y$ , and assume, without loss of generality, that  $xy \in L$  (and hence  $x'y \notin L$ ). For  $m \in \mathbb{N}$ , suppose that, when  $N$  is run on the partitioned input  $xy$  (resp.  $x'y$ ), we obtain the  $m$ -truncated crossing sequence  $Z_{m,1}, Z_{m,2}, \dots \in \widehat{\text{Den}}(\mathbb{C}^Q \otimes \mathbb{C}^C)$  (resp.  $Z'_{m,1}, Z'_{m,2}, \dots \in \widehat{\text{Den}}(\mathbb{C}^Q \otimes \mathbb{C}^C)$ ). For  $c \in C$ , let  $E_c = \mathbb{1}_{\mathbb{C}^Q} \otimes |c\rangle\langle c| \in \text{L}(\mathbb{C}^Q \otimes \mathbb{C}^C)$ . For  $s \in \mathbb{N}_{\geq 1}$ , define  $p_{m,s}, p'_{m,s} : C \rightarrow [0, 1]$  such that  $p_{m,s}(c) = \text{Tr}(E_c Z_{m,s} E_c^\dagger)$  and  $p'_{m,s}(c) = \text{Tr}(E_c Z'_{m,s} E_c^\dagger)$ . Then, for any  $c \in C$ , Lemma 13 implies

$$\begin{aligned} |p_{m,s}(c) - p'_{m,s}(c)| &= |\text{Tr}(E_c Z_{m,s} E_c^\dagger) - \text{Tr}(E_c Z'_{m,s} E_c^\dagger)| = |\text{Tr}(E_c (Z_{m,s} - Z'_{m,s}) E_c^\dagger)| \\ &\leq \|Z_{m,s} - Z'_{m,s}\|_1 \leq \frac{s-1}{2} \|N_{x,m}^{\leftarrow} - N_{x',m}^{\leftarrow}\|_1. \end{aligned}$$

Notice that  $p_{m,s}(c_{\text{acc}})$  (resp.  $p'_{m,s}(c_{\text{acc}})$ ) is the probability that  $N$  accepts  $xy$  (resp.  $x'y$ ) within the first  $s$  times (on a given branch of the computation) the head of  $N$  crosses the boundary between  $x$  (resp.  $x'$ ) and  $y$ , where any branch that runs for more than  $m$  steps between consecutive boundary crossings is forced to halt and reject immediately before attempting to perform the  $m+1^{\text{st}}$  such step. Let  $p_N(w)$  denote the probability that  $N$  accepts an input  $w \in \Sigma^*$ , let  $p_N(w, s)$  denote the probability that  $N$  accepts  $w$  within  $s$  steps, and let  $h_N(w, s)$  denote the probability that  $N$  halts on input  $w$  within  $s$  steps.

Note that  $x'y \notin L$  implies  $p_N(x'y) \leq \epsilon$ . Clearly,  $p'_{m,s}(c_{\text{acc}}) \leq p_N(x'y)$ , for any  $m$  and  $s$ , as all branches that attempt to perform more than  $m$  steps (between consecutive crossings) are considered to reject the input in the  $m$ -truncated crossing sequence. Suppose  $s \leq m$ . Any branch that runs for a total of at most  $s$  steps before halting is unaffected by  $m$ -truncation. Moreover, if a branch accepts within  $s$  steps, it will certainly accept within  $s$  crossings between  $\#_L x$  and  $y \#_R$ . This implies  $p_N(xy, s) \leq p_{m,s}(c_{\text{acc}})$ . Therefore, if  $s \leq m$ ,

$$p_N(xy, s) \leq p_{m,s}(c_{\text{acc}}) \leq p'_{m,s}(c_{\text{acc}}) + |p_{m,s}(c_{\text{acc}}) - p'_{m,s}(c_{\text{acc}})| \leq \epsilon + \frac{s-1}{2} \|N_{x,m}^{\leftarrow} - N_{x',m}^{\leftarrow}\|_1.$$

The expected running time of  $N$  on input  $xy$  is at most  $T(|xy|)$ . By Markov's inequality,  $1 - h_N(xy, s) \leq \frac{T(|xy|)}{s}$ . Note that  $xy \in L$  implies  $p_N(xy) \geq 1 - \epsilon$ . Thus, for any  $m \geq s \geq 1$ ,

$$1 - \epsilon \leq p_N(xy) \leq p_N(xy, s) + (1 - h_N(xy, s)) \leq \epsilon + \frac{s-1}{2} \|N_{x,m}^{\leftarrow} - N_{x',m}^{\leftarrow}\|_1 + \frac{T(|xy|)}{s}.$$

Set  $s = \lceil \frac{2}{1-2\epsilon} T(n) \rceil$ , and notice that  $|xy| \leq n$  implies  $T(|xy|) \leq T(n)$ . For any  $m \geq s$ ,

$$1-2\epsilon \leq \frac{\lceil \frac{2}{1-2\epsilon} T(n) \rceil - 1}{2} \|N_{x,m}^{\leftarrow} - N_{x',m}^{\leftarrow}\|_1 + \frac{T(|xy|)}{\lceil \frac{2}{1-2\epsilon} T(n) \rceil} \leq \frac{T(n)}{1-2\epsilon} \|N_{x,m}^{\leftarrow} - N_{x',m}^{\leftarrow}\|_1 + \frac{1-2\epsilon}{2}.$$

Therefore,  $T(n) \geq \frac{(1-2\epsilon)^2}{2} \|N_{x,m}^{\leftarrow} - N_{x',m}^{\leftarrow}\|_1^{-1}$ ,  $\forall m \geq \left\lceil \frac{2}{1-2\epsilon} T(n) \right\rceil$ .  $\blacktriangleleft$

**► Lemma 15.** *Consider a 2QCF A  $N = (Q, C, \Sigma, R, \theta, \delta, q_{start}, c_{start}, c_{acc}, c_{rej})$ . Let  $k = |Q|$  and  $d = |C|$ . Consider any finite  $X \subseteq \Sigma^*$  such that  $|X| \geq 2$ . Then  $\forall m \in \mathbb{N}$ ,  $\exists x, x' \in X$  such that  $x \neq x'$  and  $\|N_{x,m}^{\leftarrow} - N_{x',m}^{\leftarrow}\|_1 \leq 4\sqrt{2}k^4d^2 \left(|X|^{\frac{1}{k^4d^2}} - 1\right)^{-1}$ .*

**Proof.** For  $q, q' \in Q$  and  $c, c' \in C$ , let  $F_{q,q',c,c'} = |q\rangle\langle q'| \otimes |c\rangle\langle c'| \in L(\mathbb{C}^Q \otimes \mathbb{C}^C)$ . Let  $J : T(\mathbb{C}^Q \otimes \mathbb{C}^C) \rightarrow L(\mathbb{C}^Q \otimes \mathbb{C}^C \otimes \mathbb{C}^Q \otimes \mathbb{C}^C)$  denote the *Choi isomorphism*, which is given by  $J(\Phi) = \sum_{(q,q',c,c') \in Q^2 \times C^2} F_{q,q',c,c'} \otimes \Phi(F_{q,q',c,c'})$ ,  $\forall \Phi \in T(\mathbb{C}^Q \otimes \mathbb{C}^C)$ . Consider any  $x \in \Sigma^*$  and  $m \in \mathbb{N}$ . We first show that, if  $(c_1, c_2) \neq (c'_1, c'_2)$ , then  $\langle q_2 c_2 | N_{x,m}^{\leftarrow}(F_{q_1, q'_1, c_1, c'_1}) | q'_2 c'_2 \rangle = 0$ . To see this, recall that, by Definition 10,  $N_{x,m}^{\leftarrow} = \text{Tr}_{\mathbb{C}^{H_x}} \circ T_x \circ S_x^m \circ I_x$ . If  $c_1 \neq c'_1$ , then  $N_{x,m}^{\leftarrow}(F_{q_1, q'_1, c_1, c'_1}) = 0_{\mathbb{C}^Q \otimes \mathbb{C}^C}$ , which implies  $\langle q_2 c_2 | N_{x,m}^{\leftarrow}(F_{q_1, q'_1, c_1, c'_1}) | q'_2 c'_2 \rangle = 0$ . If  $c_2 \neq c'_2$ , then  $\forall Z \in L(\mathbb{C}^Q \otimes \mathbb{C}^C \otimes \mathbb{C}^{H_x})$ ,  $\langle q_2 c_2 | \text{Tr}_{\mathbb{C}^{H_x}}(T_x(Z)) | q'_2 c'_2 \rangle = 0$ , which implies  $\langle q_2 c_2 | N_{x,m}^{\leftarrow}(F_{q_1, q'_1, c_1, c'_1}) | q'_2 c'_2 \rangle = 0$ .

Therefore,  $\langle q_2 c_2 | N_{x,m}^{\leftarrow}(F_{q_1, q'_1, c_1, c'_1}) | q'_2 c'_2 \rangle$  is only potentially non-zero at the  $k^4d^2$  elements where  $(c_1, c_2) = (c'_1, c'_2)$ . By Lemma 11(iii),  $N_{x,m}^{\leftarrow} \in \text{Chan}(\mathbb{C}^Q \otimes \mathbb{C}^C)$ , which implies  $J(N_{x,m}^{\leftarrow}) \in \text{Pos}(\mathbb{C}^Q \otimes \mathbb{C}^C \otimes \mathbb{C}^Q \otimes \mathbb{C}^C)$  [37, Corollary 2.27]. Therefore, the elements where  $(q_1, q_2) \neq (q'_1, q'_2)$  come in conjugate pairs, and the elements with  $(q_1, q_2) = (q'_1, q'_2)$  are real. We define the function  $g_{N,m} : \Sigma^* \rightarrow \mathbb{R}^{k^4d^2}$  such that  $g_{N,m}(x)$  encodes all the potentially non-zero  $\langle q_2 c_2 | N_{x,m}^{\leftarrow}(F_{q_1, q'_1, c_1, c'_1}) | q'_2 c'_2 \rangle$ , without redundancy (only encoding one element of a conjugate pair). To be precise, the first  $k^2d^2$  entries of  $g_{N,m}(x)$  are given by  $\{\langle q_2 c_2 | N_{x,m}^{\leftarrow}(F_{q_1, q_1, c_1, c_1}) | q_2 c_2 \rangle : q_1, q_2 \in Q, c_1, c_2 \in C\} \subseteq \mathbb{R}$ . Establish some total order  $\geq$  on  $Q$ , and let  $\widehat{Q}^4 = \{(q_1, q'_1, q_2, q'_2) \in Q^4 : q'_1 > q_1 \text{ or } (q'_1 = q_1 \text{ and } q'_2 > q_2)\}$ . The remaining  $k^4d^2 - k^2d^2$  entries are given by encoding each of the  $\frac{1}{2}(k^4d^2 - k^2d^2)$  potentially non-zero entries  $\{\langle q_2 c_2 | N_{x,m}^{\leftarrow}(F_{q_1, q'_1, c_1, c_1}) | q'_2 c_2 \rangle : (q_1, q'_1, q_2, q'_2) \in \widehat{Q}^4, c_1, c_2 \in C\} \subseteq \mathbb{C}$  as the pair of real numbers that comprise their real and imaginary parts.

Let  $h = k^4d^2$ . Let  $\|\cdot\| : \mathbb{R}^h \rightarrow \mathbb{R}_{\geq 0}$  denote the Euclidean 2-norm and  $\|\cdot\|_2 : L(V) \rightarrow \mathbb{R}_{\geq 0}$  denote the Schatten 2-norm. Note that  $\|\Phi\|_1 \leq \|J(\Phi)\|_1$ ,  $\forall \Phi$  [37, Section 3.4]. We have,

$$\begin{aligned} \|N_{x,m}^{\leftarrow} - N_{x',m}^{\leftarrow}\|_1 &\leq \|J(N_{x,m}^{\leftarrow} - N_{x',m}^{\leftarrow})\|_1 \leq \sqrt{\text{rank}(J(N_{x,m}^{\leftarrow} - N_{x',m}^{\leftarrow}))} \|J(N_{x,m}^{\leftarrow} - N_{x',m}^{\leftarrow})\|_2 \\ &\leq \sqrt{h} \|J(N_{x,m}^{\leftarrow}) - J(N_{x',m}^{\leftarrow})\|_2 \leq \sqrt{2h} \|g_{N,m}(x) - g_{N,m}(x')\|. \end{aligned}$$

Note that  $N_{x,m}^{\leftarrow} \in \text{Chan}(\mathbb{C}^Q \otimes \mathbb{C}^C)$ , which implies  $\|N_{x,m}^{\leftarrow}\|_1 = 1$  [37, Corollary 3.40]. Then,  $\forall q, q' \in Q, \forall c \in C$ , we have  $\|F_{q,q',c,c}\|_1 = 1$ , which implies  $\|N_{x,m}^{\leftarrow}(F_{q,q',c,c})\|_1 \leq 1$ . Therefore,

$$\|g_{N,m}(x)\| \leq \|J(N_{x,m}^{\leftarrow})\|_2 \leq \|J(N_{x,m}^{\leftarrow})\|_1 \leq \sum_{q,q' \in Q, c \in C} \|N_{x,m}^{\leftarrow}(F_{q,q',c,c})\|_1 \leq k^2d = \sqrt{h}.$$

For  $v_0 \in \mathbb{R}^h$  and  $r \in \mathbb{R}_{>0}$ , let  $B(v_0, r) = \{v \in \mathbb{R}^h : \|v_0 - v\| \leq r\}$  denote the closed ball centered at  $v_0$  of radius  $r$  in  $\mathbb{R}^h$ , which has volume  $\text{vol}(B(v_0, r)) = c_h r^h$ , for some constant  $c_h \in \mathbb{R}_{>0}$ . By the above,  $\|g_{N,m}(x)\| \leq \sqrt{h}$ , which implies that  $B(g_{N,m}(x), \delta) \subseteq B(0, \sqrt{h} + \delta)$ ,  $\forall \delta \in \mathbb{R}_{>0}$ . Suppose  $\forall x, x' \in X$  with  $x \neq x'$ , we have  $B(g_{N,m}(x), \delta) \cap B(g_{N,m}(x'), \delta) = \emptyset$ .



Then  $\sqcup_{x \in X} B(g_{N,m}(x), \delta) \subseteq B(0, \sqrt{h} + \delta)$ , which implies  $|X|c_h\delta^h \leq c_h(\sqrt{h} + \delta)^h$ . Set  $\delta = \frac{2\sqrt{h}}{|X|^{1/h} - 1}$ . Then  $\exists x, x' \in X$ , with  $x \neq x'$ , such that  $B(g_{N,m}(x), \delta) \cap B(g_{N,m}(x'), \delta) \neq \emptyset$ , which implies  $\|g_{N,m}(x) - g_{N,m}(x')\| \leq 2\delta$ . Therefore,

$$\|N_{x,m}^{\leftarrow} - N_{x',m}^{\leftarrow}\|_1 \leq \sqrt{2h}\|g_{N,m}(x) - g_{N,m}(x')\| \leq \sqrt{2h}2\delta \leq 4\sqrt{2}k^4d^2 \left(|X|^{\frac{1}{k^4d^2}} - 1\right)^{-1}. \blacktriangleleft$$

We now prove a 2QCFA analogue of the Dwork-Stockmeyer lemma.

► **Theorem 16.** *If  $L \in \text{B2QCFA}(k, d, T(n), \epsilon)$ , for some  $k, d \in \mathbb{N}_{\geq 2}$ ,  $T : \mathbb{N} \rightarrow \mathbb{N}$ , and  $\epsilon \in [0, \frac{1}{2})$ , then  $\exists N_0 \in \mathbb{N}$  such that  $T(n) \geq \frac{(1-2\epsilon)^2}{16\sqrt{2}k^4d^2} D_L(n)^{\frac{1}{k^4d^2}}$ ,  $\forall n \geq N_0$ .*

**Proof.** Consider some  $L \subseteq \Sigma^*$ . By [10, Lemma 3.1],  $L \in \text{REG} \Leftrightarrow \exists b \in \mathbb{N}_{\geq 1}$  such that  $D_L(n) \leq b$ ,  $\forall n \in \mathbb{N}$ . Thus, if  $L \in \text{REG}$ , the claim is immediate (recall that  $T(n) \geq n$ ). Next, suppose  $L \notin \text{REG}$ . For  $n \in \mathbb{N}$ , define  $X_n = \{x_1, \dots, x_{D_L(n)}\} \subseteq \Sigma^{\leq n}$  such that the  $x_i$  are pairwise  $(L, n)$ -dissimilar. As  $D_L(n)$  is not bounded above by any constant,  $\exists N_0 \in \mathbb{N}$  such that  $D_L(N_0) \geq 2^{k^4d^2}$ . Then,  $\forall n \geq N_0$ , we have  $|X_n| = D_L(n) \geq D_L(N_0) \geq 2^{k^4d^2}$ . Fix  $n \geq N_0$  and set  $m = \lceil \frac{1-2\epsilon}{2} T(n) \rceil$ . By Lemma 15,  $\exists x, x' \in X_n$  such that  $x \neq x'$  and

$$\|N_{x,m}^{\leftarrow} - N_{x',m}^{\leftarrow}\|_1 \leq 4\sqrt{2}k^4d^2 \left(|X_n|^{\frac{1}{k^4d^2}} - 1\right)^{-1} \leq 8\sqrt{2}k^4d^2 |X_n|^{-\frac{1}{k^4d^2}} = 8\sqrt{2}k^4d^2 D_L(n)^{-\frac{1}{k^4d^2}}.$$

Fix such a pair  $x, x'$ , and note that  $x \not\sim_{L,n} x'$ , by construction. By Lemma 14,

$$T(n) \geq \frac{(1-2\epsilon)^2}{2} \|N_{x,m}^{\leftarrow} - N_{x',m}^{\leftarrow}\|_1^{-1} \geq \frac{(1-2\epsilon)^2}{16\sqrt{2}k^4d^2} D_L(n)^{\frac{1}{k^4d^2}}. \blacktriangleleft$$

### 4.3 2QCFA Running Time Lower Bounds and Complexity Class Separations

Let  $\text{B2QCFA}(T(n)) = \cup_{k,d \in \mathbb{N}_{\geq 2}, \epsilon \in [0, \frac{1}{2})} \text{B2QCFA}(k, d, T(n), \epsilon)$  denote the class of languages recognizable with two-sided bounded error by a 2QCFA with any constant number of quantum and classical states, in expected time at most  $T(n)$ . For a family  $\mathcal{T}$  of functions of the form  $T : \mathbb{N} \rightarrow \mathbb{N}$ , let  $\text{B2QCFA}(\mathcal{T}) = \cup_{T \in \mathcal{T}} \text{B2QCFA}(T(n))$ . We then write, for example,  $\text{B2QCFA}(2^{o(n)})$  to denote the union, taken over every function  $T : \mathbb{N} \rightarrow \mathbb{N}$  such that  $T(n) = 2^{o(n)}$ , of  $\text{B2QCFA}(T(n))$ . Let  $C_L : \mathbb{N} \rightarrow \mathbb{N}$  denote the *one-way deterministic communication complexity* of testing membership in  $L$ ; note that  $C_L(n) = \log D_L(n)$ ,  $\forall n$  [8]. We immediately obtain the following corollaries of Theorem 16.

► **Corollary 17.** *If  $L \in \text{B2QCFA}(T(n))$ , then  $D_L(n) = T(n)^{O(1)}$  and  $C_L(n) = O(\log T(n))$ .*

► **Corollary 18.** *If a language  $L$  satisfies  $D_L(n) = 2^{\Omega(n)}$ , then  $L \notin \text{B2QCFA}(2^{o(n)})$ .*

Notice that  $D_L(n) = 2^{O(n)}$ , for any  $L$ . We next exhibit a language for which  $D_L(n) = 2^{\Omega(n)}$ , thereby yielding a strong lower bound on the running time of any 2QCFA that recognizes  $L$ . For  $w = w_1 \dots w_n \in \Sigma^*$ , let  $w^{\text{rev}} = w_n \dots w_1$  denote the reversal of the string  $w$ . Let  $L_{\text{pal}} = \{w \in \{a, b\}^* : w = w^{\text{rev}}\}$  consist of all palindromes over the alphabet  $\{a, b\}$ .

► **Corollary 19.**  $L_{\text{pal}} \notin \text{B2QCFA}(2^{o(n)})$ .

**Proof.** For  $n \in \mathbb{N}$ , let  $W_n = \{w \in \{a, b\}^* : |w| = n\}$  denote all words over the alphabet  $\{a, b\}$  of length  $n$ . For any  $w, w' \in W_n$ , with  $w \neq w'$ , we have  $|ww^{\text{rev}}| = 2n = |w'w^{\text{rev}}|$ ,  $ww^{\text{rev}} \in L_{\text{pal}}$ , and  $w'w^{\text{rev}} \notin L_{\text{pal}}$ ; therefore,  $w \not\sim_{L_{\text{pal}}, 2n} w'$ ,  $\forall w, w' \in W_n$  such that  $w \neq w'$ . This implies that  $D_{L_{\text{pal}}}(2n) \geq |W_n| = 2^n$ . Corollary 18 then implies  $L_{\text{pal}} \notin \text{B2QCFA}(T(n))$ . ◀



We define  $\text{BQE2QCFA} = \text{B2QCFA}(2^{O(n)})$  to be the class of languages recognizable with two-sided bounded error in expected exponential time (with linear exponent) by a 2QCFA. Next, we say that a 2QCFA  $N$  recognizes a language  $L$  with *negative one-sided bounded error*  $\epsilon \in \mathbb{R}_{>0}$  if,  $\forall w \in L$ ,  $\Pr[N \text{ accepts } w] = 1$ , and,  $\forall w \notin L$ ,  $\Pr[N \text{ accepts } w] \leq \epsilon$ . We define  $\text{coR2QCFA}(k, d, T(n), \epsilon)$  as the class of languages recognizable with negative one-sided bounded error  $\epsilon$  by a 2QCFA, with at most  $k$  quantum basis states and at most  $d$  classical states, that has expected running time at most  $T(n)$  on all inputs of length at most  $n$ . We define  $\text{coR2QCFA}(T(n))$  and  $\text{coRQE2QCFA}$  analogously to the two-sided bounded error case.

Ambainis and Watrous [2] showed that  $L_{\text{pal}} \in \text{coRQE2QCFA}$ ; in fact, their 2QCFA recognizer for  $L_{\text{pal}}$  has only a single-qubit. Clearly,  $\text{coR2QCFA}(T(n)) \subseteq \text{B2QCFA}(T(n))$ , for any  $T$ , and  $\text{coRQE2QCFA} \subseteq \text{BQE2QCFA}$ . Therefore, the class of languages recognizable by a 2QCFA with bounded error in *subexponential* time is properly contained in the class of languages recognizable by a 2QCFA in *exponential* time.

► **Corollary 20.**  $\text{B2QCFA}(2^{o(n)}) \subsetneq \text{BQE2QCFA}$  and  $\text{coR2QCFA}(2^{o(n)}) \subsetneq \text{coRQE2QCFA}$ .

We next define  $\text{BQP2QCFA} = \text{B2QCFA}(n^{O(1)})$  to be the class of languages recognizable with two-sided bounded error in expected polynomial time by a 2QCFA. See the full version [26] for a proof of the following corollary.

► **Corollary 21.** If  $L \in \text{BQP2QCFA}$ , then  $D_L(n) = n^{O(1)}$ . Therefore,  $\text{BQP2QCFA} \subseteq L/\text{poly}$ .

Of course, there are many languages  $L$  for which one can establish a strong lower bound on  $D_L(n)$ , and thereby establish a strong lower bound on the expected running time  $T(n)$  of any 2QCFA that recognizes  $L$ . In Section 6, we consider the case in which  $L$  is the word problem of a group, and we show that very strong lower bounds can be established on  $D_L(n)$ . In the current section, we consider two especially interesting languages; the relevance of these languages was brought to our attention by Richard Lipton (personal communication). For  $p \in \mathbb{N}$ , let  $\langle p \rangle_2 \in \{0, 1\}^*$  denote its binary representation; let  $L_{\text{primes}} = \{\langle p \rangle_2 : p \text{ is prime}\}$ . Note that  $D_{L_{\text{primes}}}(n) = 2^{\Omega(n)}$  [29], which immediately implies the following.

► **Corollary 22.**  $L_{\text{primes}} \notin \text{B2QCFA}(2^{o(n)})$ .

Say a string  $w = w_1 \cdots w_n \in \{0, 1\}^n$  has a length-3 arithmetic progression (3AP) if  $\exists i, j, k \in \mathbb{N}$  such that  $1 \leq i < j < k \leq n$ ,  $j - i = k - j$ , and  $w_i = w_j = w_k = 1$ ; let  $L_{3\text{ap}} = \{w \in \{0, 1\}^* : w \text{ has a 3AP}\}$ . It is straightforward to show the lower bound  $D_{L_{3\text{ap}}}(n) = 2^{n^{1-o(1)}}$ , as well as the upper bound  $D_{L_{3\text{ap}}}(n) = 2^{n^{o(n)}}$ . Therefore, one obtains the following lower bound on the running time of a 2QCFA that recognizes  $L_{3\text{ap}}$ , which, while still quite strong, is not as strong as that of  $L_{\text{pal}}$  or  $L_{\text{primes}}$ .

► **Corollary 23.**  $L_{3\text{ap}} \notin \text{B2QCFA}(2^{n^{1-\Omega(1)}})$ .

► **Remark.** While  $L_{\text{primes}}$  and  $L_{3\text{ap}}$  provide two more examples of natural languages for which our method yields strong lower bound on the running time of any 2QCFA recognizer, they also suggest the potential of proving a stronger lower bound for certain languages. That is to say, for  $L_{\text{pal}}$ , one has (essentially) matching lower and upper bounds on the running time of any 2QCFA recognizer; this is certainly not the case for  $L_{\text{primes}}$  and  $L_{3\text{ap}}$ . In fact, we currently do not know if either  $L_{\text{primes}}$  or  $L_{3\text{ap}}$  can be recognized by a 2QCFA with bounded error *at all* (i.e., regardless of time bound).

#### 4.4 Transition Amplitudes of 2QCFA

As in Definition 3, for some 2QCFA  $N = (Q, C, \Sigma, R, \theta, \delta, q_{\text{start}}, c_{\text{start}}, c_{\text{acc}}, c_{\text{rej}})$ , let  $\{E_{c,\sigma,r,j} : r \in R, j \in J\} \subseteq L(\mathbb{C}^Q)$  denote the set of operators that describe the selective quantum operation  $\theta(c, \sigma) \in \text{QuantOp}(\mathbb{C}^Q, R)$  that is applied to the quantum register when the classical state of  $N$  is  $c \in \hat{C}$  and the head of  $N$  is over the symbol  $\sigma \in \Sigma_+$ . The *transition amplitudes* of  $N$  are the set of numbers  $\{\langle q | E_{c,\sigma,r,j} | q' \rangle : c \in \hat{C}, \sigma \in \Sigma_+, r \in R, j \in J, q, q' \in Q\} \subseteq \mathbb{C}$ .

While other types of finite automata are often defined without any restriction on their transition amplitudes, for 2QCFA, and other types of QFA, the allowed class of transition amplitudes strongly affects the power of the model. For example, using non-computable transition amplitudes, a 2QCFA can recognize certain undecidable languages with bounded error in expected polynomial time [28]. Our lower bound holds even in this setting of unrestricted transition amplitudes. For  $\mathbb{F} \subseteq \mathbb{C}$ , we define complexity classes  $\text{coR2QCFA}_{\mathbb{F}}(k, d, T(n), \epsilon)$ ,  $\text{coRQE2QCFA}_{\mathbb{F}}$ , etc., that are variants of the corresponding complexity class in which the 2QCFA are restricted to have transition amplitudes in  $\mathbb{F}$ . Using our terminology, Ambainis and Watrous [2] showed that  $L_{\text{pal}} \in \text{coRQE2QCFA}_{\overline{\mathbb{Q}}}$ , where  $\overline{\mathbb{Q}}$  denotes the algebraic numbers, which are, arguably, the natural choice for the permitted class of transition amplitudes of a quantum model of computation. Therefore,  $L_{\text{pal}}$  can be recognized with negative one-sided bounded error by a single-qubit 2QCFA with transition amplitudes that are all algebraic numbers in expected exponential time; however,  $L_{\text{pal}}$  cannot be recognized with two-sided bounded error (and, therefore, not with one-sided bounded error) by a 2QCFA (of any constant size) in subexponential time, regardless of the permitted transition amplitudes.

### 5 Lower Bounds on the Running Time of Small-Space QTM

We next show that our technique also yields a lower bound on the expected running time of a quantum Turing machine (QTM) that uses sublogarithmic space (i.e.,  $o(\log n)$  space). The key idea is that a QTM  $M$  that uses  $S(n)$  space can be viewed as a sequence  $(M_n)_{n \in \mathbb{N}}$  of 2QCFA, where  $M_n$  has  $2^{O(S(n))}$  (classical and quantum) states and  $M_n$  simulates  $M$  on all inputs of length at most  $n$  (therefore,  $M_n$  and  $M$  have the same probability of acceptance and the same expected running time on any such input). The techniques of the previous section apply to 2QCFA with a sufficiently slowly growing number of states.

We consider the *classically controlled* space-bounded QTM model that allows *intermediate measurements*, following the definition of Watrous [35]. While several such QTM models have been defined, we focus on this model as we wish to prove our lower bound in the greatest generality possible. We note that the definitions of such QTM models by, for instance, Ta-Shma [32], Watrous [36, Section VII.2], and (essentially, without the use of random access) van Melkebeek and Watson [22] are special cases of the QTM model that we consider. In the case of time-bounded quantum computation, it is well-known that allowing a QTM to perform intermediate measurements provably does not increase the power of the model; very recently, this fact has also been shown to hold in the simultaneously time-bounded and space-bounded setting [12]. Let  $\text{BQTISP}(T(n), S(n))$  denote the class of languages recognizable with bounded error by a QTM in time  $T(n)$  and space  $S(n)$ . See the full version [26] for a complete definition of the QTM model and a proof of the following theorem.

► **Theorem 24.** *Suppose  $L \in \text{BQTISP}(T(n), S(n))$ , and suppose further that  $S(n) = o(\log \log D_L(n))$ . Then  $\exists b_0 \in \mathbb{R}_{>0}$  such that,  $T(n) = \Omega(2^{-b_0 S(n)} D_L(n)^{2^{-b_0 S(n)}})$ .*

► **Corollary 25.** *If  $D_L(n) = 2^{\Omega(n)}$ , then  $L \notin \text{BQTISP}(2^{n^{1-\Omega(1)}}, o(\log n))$ . In particular,  $L_{\text{pal}} \notin \text{BQTISP}(2^{n^{1-\Omega(1)}}, o(\log n))$ .*

► **Remark.** Of course,  $L_{pal}$  can be recognized by a *deterministic* TM in  $O(\log n)$  space (and, trivially, polynomial time). Therefore, the previous corollary exhibits a natural problem for which polynomial time *quantum* TM cannot outperform polynomial time *deterministic* TM in terms of the amount of space used.

## 6 The Word Problem of a Group

We begin by formally defining the word problem of a group; for further background, see, for instance [21]. For a set  $S$ , let  $F(S)$  denote the free group on  $S$ . For sets  $S, R$  such that  $R \subseteq F(S)$ , let  $N$  denote the normal closure of  $R$  in  $F(S)$ ; for a group  $G$ , if  $G \cong F(S)/N$ , then we say that  $G$  has *presentation*  $\langle S|R \rangle$ , which we denote by writing  $G = \langle S|R \rangle$ . Suppose  $G = \langle S|R \rangle$ , with  $S$  finite; we now define  $W_{G=\langle S|R \rangle}$ , the *word problem of  $G$  with respect to the presentation  $\langle S|R \rangle$* . We define the set of formal inverses  $S^{-1}$ , such that, for each  $s \in S$ , there is a unique corresponding  $s^{-1} \in S^{-1}$ , and  $S \cap S^{-1} = \emptyset$ . Let  $\Sigma = S \sqcup S^{-1}$ , let  $\Sigma^*$  denote the free monoid over  $\Sigma$ , and let  $\phi : \Sigma^* \rightarrow G$  be the natural (monoid) homomorphism that takes each string in  $\Sigma^*$  to the element of  $G$  that it represents. We use  $1_G$  to denote the identity element of  $G$ . Then  $W_{G=\langle S|R \rangle} = \phi^{-1}(1_G)$ . Note that the definition of the word problem does depend on the choice presentation. However, if  $\mathcal{L}$  is any complexity class that is closed under inverse homomorphism, then if  $\langle S|R \rangle$  and  $\langle S'|R' \rangle$  are both presentations of some group  $G$ , and  $S$  and  $S'$  are both finite, then  $W_{G=\langle S|R \rangle} \in \mathcal{L} \Leftrightarrow W_{G=\langle S'|R' \rangle} \in \mathcal{L}$  [19]. As all complexity classes considered in this paper are easily seen to be closed under inverse homomorphism, we will simply write  $W_G \in \mathcal{L}$  to mean that  $W_{G=\langle S|R \rangle} \in \mathcal{L}$ , for every presentation  $G = \langle S|R \rangle$ , with  $S$  finite. We note that the languages  $L_{pal}$  and  $L_{eq}$ , which Ambainis and Watrous [2] showed satisfy  $L_{pal} \in \text{coRQE2QCFA}_{\mathbb{Q}}$  and  $L_{eq} \in \text{BQP2QCFA}$ , are closely related to the word problems of the groups  $F_2$  and  $\mathbb{Z}$ , respectively.

### 6.1 The Growth Rate of a Group and Nonregularity

Consider a group  $G = \langle S|R \rangle$ , with  $S$  finite. Define  $\Sigma$  and  $\phi$  as in the previous section. For  $g \in G$ , let  $l_S(g)$  denote the smallest  $m \in \mathbb{N}$  such that  $\exists \sigma_1, \dots, \sigma_m \in \Sigma$  such that  $g = \phi(\sigma_1 \cdots \sigma_m)$ . For  $n \in \mathbb{N}$ , we define  $B_{G,S}(n) = \{g \in G : l_S(g) \leq n\}$  and we further define  $\beta_{G,S}(n) = |B_{G,S}(n)|$ , which we call the *growth rate of  $G$  with respect to  $S$* . The following straightforward lemma demonstrates an important relationship between  $\beta_{G,S}$  and  $D_{W_{G=\langle S|R \rangle}}$ .

► **Lemma 26.** *Suppose  $G = \langle S|R \rangle$  with  $S$  finite. Using the notation established above, let  $W_G := W_{G=\langle S|R \rangle} = \phi^{-1}(1_G)$  denote the word problem of  $G$  with respect to this presentation. Then,  $\forall n \in \mathbb{N}$ ,  $D_{W_G}(2n) \geq \beta_{G,S}(n)$ .*

**Proof.** Fix  $n \in \mathbb{N}$ , let  $k = \beta_{G,S}(n)$ , and let  $B_{G,S}(n) = \{g_1, \dots, g_k\}$ . For a string  $x = x_1 \cdots x_m \in \Sigma^*$ , where each  $x_j \in \Sigma$ , let  $|x| = m$  denote the (string) length of  $x$  and define  $x^{-1} = x_m^{-1} \cdots x_1^{-1}$ . Note that,  $\forall g \in G$ ,  $l_S(g) = \min_{w \in \phi^{-1}(g)} |w|$ . Therefore, for each  $i \in \{1, \dots, k\}$  we may define  $w_i \in \phi^{-1}(g_i)$  such that  $|w_i| = l_S(g_i)$ . Observe that  $w_i w_i^{-1} \in W_G$  and  $|w_i w_i^{-1}| = 2|w_i| = 2l_S(g_i) \leq 2n$ ; moreover, for each  $j \neq i$ , we have  $w_j w_i^{-1} \notin W_G$  and  $|w_j w_i^{-1}| = |w_j| + |w_i| = l_S(g_j) + l_S(g_i) \leq 2n$ . Therefore,  $w_1, \dots, w_k$  are pairwise  $(W_G, 2n)$ -dissimilar, which implies  $D_{W_G}(2n) \geq k = \beta_{G,S}(n)$ . ◀

For a pair of non-decreasing functions  $f_1, f_2 : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$ , we write  $f_1 \prec f_2$  if  $\exists C_1, C_2 \in \mathbb{R}_{>0}$  such that  $\forall r \in \mathbb{R}_{\geq 0}$ ,  $f_1(r) \leq C_1 f_2(C_1 r + C_2) + C_2$ ; we write  $f_1 \sim f_2$  if both  $f_1 \prec f_2$  and  $f_2 \prec f_1$ . Suppose  $\langle S|R \rangle$  and  $\langle S'|R' \rangle$  are both presentations of  $G$ , with  $S$  and  $S'$  finite. It is straightforward to show that  $\beta_{G,S}$  and  $\beta_{G,S'}$  are non-decreasing, and that  $\beta_{G,S} \sim \beta_{G,S'}$  [21, Proposition 6.2.4]. For this reason, we will simply write  $\beta_G$  to denote the growth rate of  $G$ .

► **Definition 27.** Suppose  $G$  is a finitely generated group. If  $\beta_G \sim (n \mapsto e^n)$ , we say  $G$  has *exponential growth*. If  $\exists c \in \mathbb{R}_{\geq 0}$  such that  $\beta_G \prec (n \mapsto n^c)$ , we say  $G$  has *polynomial growth*. Otherwise, we say  $G$  has *intermediate growth*. Note that, for any finitely generated group  $G$ , we have  $\beta_G \prec (n \mapsto e^n)$ , and so the term “intermediate growth” is justified.

## 6.2 Word Problems Recognizable by 2QCFA and Small-Space QTM

By making use of two very powerful results in group theory, the Tits’ Alternative [34] and Gromov’s theorem on groups of polynomial growth [15], we exhibit useful lower bounds on  $D_{W_G}$ , which in turn allows us to show a strong lower bound on the expected running time of a 2QCFA that recognizes  $W_G$ . We obtain an analogous result for sublogarithmic-space QTM; see the full paper [26] for details.

► **Theorem 28.** *For any finitely generated group  $G$ , the following statements hold.*

- (i) *If  $W_G \in \text{B2QCFA}(k, d, T(n), \epsilon)$ , then  $\beta_G \prec (n \mapsto T(n)^{k^4 d^2})$ .*
- (ii) *If  $G$  has exponential growth, then  $W_G \notin \text{B2QCFA}(2^{o(n)})$ .*
- (iii) *If  $G$  is a linear group over a field of characteristic 0, and  $G$  is not virtually nilpotent, then  $W_G \notin \text{B2QCFA}(2^{o(n)})$ .*
- (iv) *If  $W_G \in \text{BQP2QCFA}$ , then  $G$  is virtually nilpotent.*

**Proof.**

- (i) Follows immediately from Lemma 26 and Corollary 17.
- (ii) Follows immediately from Definition 27 and part (i) of this theorem.
- (iii) As a consequence of the famous Tits’ Alternative [34], every finitely generated linear group over a field of characteristic 0 either has polynomial growth or exponential growth, and has polynomial growth precisely when it is virtually nilpotent ([34, Corollary 1], [38]). The claim then follows by part (ii) of this theorem.
- (iv) If  $W_G \in \text{BQP2QCFA}$ , then  $W_G \in \text{B2QCFA}(k, d, n^c, \epsilon)$  for some  $k, d, c \in \mathbb{N}_{\geq 1}, \epsilon \in [0, \frac{1}{2})$ . By part (i) of this theorem,  $\beta_G \prec (n \mapsto n^{ck^4 d^2})$ , which implies  $G$  has polynomial growth. By Gromov’s theorem on groups of polynomial growth [15], a finitely generated group has polynomial growth precisely when it is virtually nilpotent. ◀

► **Remark.** All *known*  $G$  of intermediate growth have  $\beta_G \sim (n \mapsto e^{n^c})$ , for some  $c \in (1/2, 1)$ . Therefore, a strong lower bound may be established on the running time of any 2QCFA that recognizes  $W_G$ , for any known group of intermediate growth.

Let  $\mathcal{G}_{\mathbf{vAb}}$  (resp.  $\mathcal{G}_{\mathbf{vNilp}}$ ) denote the collection of all finitely generated virtually abelian (resp. nilpotent) groups. Let  $U(k, \overline{\mathbb{Q}})$  denote the group of  $k \times k$  unitary matrices with algebraic number entries, and let  $\mathcal{U}$  consist of all finitely generated subgroups of any  $U(k, \overline{\mathbb{Q}})$ . We have recently shown that if  $G \in \mathcal{U}$ , then  $W_G \in \text{coRQE2QCFA}_{\overline{\mathbb{Q}}}$  [27, Corollary 1.4.1]. Observe that  $\mathcal{G}_{\mathbf{vAb}} \subseteq \mathcal{U}$  and that all groups in  $\mathcal{U}$  are finitely generated linear groups over a field of characteristic zero. Moreover,  $\mathcal{U} \cap \mathcal{G}_{\mathbf{vNilp}} = \mathcal{G}_{\mathbf{vAb}}$  [33, Proposition 2.2]. We, therefore, obtain the following corollary of Theorem 28, which exhibits a broad and natural class of languages that a 2QCFA can recognize in exponential time, but not in subexponential time.

► **Corollary 29.**  $\forall G \in \mathcal{U} \setminus \mathcal{G}_{\mathbf{vAb}}$ , we have  $W_G \in \text{coRQE2QCFA}_{\overline{\mathbb{Q}}}$  but  $W_G \notin \text{B2QCFA}(2^{o(n)})$ .

We have also recently shown that  $W_G \in \text{coRQP2QCFA}_{\overline{\mathbb{Q}}}(2) \subseteq \text{BQP2QCFA}$ ,  $\forall G \in \mathcal{G}_{\mathbf{vAb}}$  [27, Theorem 1.2]. By Theorem 28, if  $W_G \in \text{BQP2QCFA}$ , then  $G \in \mathcal{G}_{\mathbf{vNilp}}$ . This naturally raises the question of whether or not there is some  $G \in \mathcal{G}_{\mathbf{vNilp}} \setminus \mathcal{G}_{\mathbf{vAb}}$  such that  $W_G \in \text{BQP2QCFA}$ . Consider the (three-dimensional discrete) Heisenberg group  $H = \langle x, y, z \mid z = [x, y], [x, z] = [y, z] = 1 \rangle$ .  $W_H$  is a natural choice for a potential “hard” word problem for 2QCFA, due

to the lack of faithful finite-dimensional unitary representations of  $H$  (see [27] for further discussion). We next show that if  $W_H \notin \text{BQP2QCFA}$ , then we have a complete classification of those word problems recognizable by 2QCFA in polynomial time.

► **Proposition 30.** *If  $W_H \notin \text{BQP2QCFA}$ , then  $W_G \in \text{BQP2QCFA} \Leftrightarrow G \in \mathcal{G}_{vAb}$ .*

**Proof.** By the above discussion, it suffices to show the following claim: if  $W_G \in \text{BQP2QCFA}$ , for some  $G \in \mathcal{G}_{vNilp} \setminus \mathcal{G}_{vAb}$ , then  $W_H \in \text{BQP2QCFA}$ . Begin by noting that  $\forall G \in \mathcal{G}_{vNilp} \setminus \mathcal{G}_{vAb}$ ,  $G$  has a subgroup isomorphic to  $H$  [20, Theorem 12]. It is straightforward to see that BQP2QCFA is closed under inverse homomorphism and intersection with regular languages. Therefore, if  $W_G \in \text{BQP2QCFA}$ , then  $W_H \in \text{BQP2QCFA}$  [20, Lemma 2]. ◀

## References

- 1 Leonard Adleman. Two theorems on random polynomial time. In *19th Annual Symposium on Foundations of Computer Science (sfcs 1978)*, pages 75–83. IEEE, 1978.
- 2 Andris Ambainis and John Watrous. Two-way finite automata with quantum and classical states. *Theoretical Computer Science*, 287(1):299–311, 2002.
- 3 Andris Ambainis and Abuzer Yakaryılmaz. Automata and quantum computing. *arXiv preprint*, 2015. [arXiv:1507.01988](#).
- 4 Ao V Anisimov. Group languages. *Cybernetics and Systems Analysis*, 7(4):594–601, 1971.
- 5 Frank Arute, Kunal Arya, Ryan Babbush, Dave Bacon, Joseph C Bardin, Rami Barends, Rupak Biswas, Sergio Boixo, Fernando GSL Brandao, David A Buell, et al. Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779):505–510, 2019.
- 6 J-C Birget, A Yu Ol’shanskii, Eliyahu Rips, and Mark V Sapir. Isoperimetric functions of groups and computational complexity of the word problem. *Annals of Mathematics*, pages 467–518, 2002.
- 7 Allan Borodin, Stephen Cook, and Nicholas Pippenger. Parallel computation for well-endowed rings and space-bounded probabilistic machines. *Information and Control*, 58(1-3), 1983.
- 8 Anne Condon, Lisa Hellerstein, Samuel Pottle, and Avi Wigderson. On the power of finite automata with both nondeterministic and probabilistic states. *SIAM Journal on Computing*, 27(3):739–762, 1998.
- 9 Martin J Dunwoody. The accessibility of finitely presented groups. *Inventiones mathematicae*, 81(3):449–457, 1985.
- 10 Cynthia Dwork and Larry Stockmeyer. A time complexity gap for two-way probabilistic finite-state automata. *SIAM Journal on Computing*, 19(6):1011–1023, 1990.
- 11 Cynthia Dwork and Larry Stockmeyer. Finite state verifiers I: The power of interaction. *Journal of the ACM (JACM)*, 39(4):800–828, 1992.
- 12 Bill Fefferman and Zachary Remschrin. Eliminating intermediate measurements in space-bounded quantum computation, 2020. [arXiv:2006.03530](#).
- 13 Rūsiņš Freivalds. Probabilistic two-way machines. In *International Symposium on Mathematical Foundations of Computer Science*, pages 33–45. Springer, 1981.
- 14 Albert G Greenberg and Alan Weiss. A lower bound for probabilistic algorithms for finite state machines. *Journal of Computer and System Sciences*, 33(1):88–105, 1986.
- 15 Michael Gromov. Groups of polynomial growth and expanding maps (with an appendix by Jacques Tits). *Publications Mathématiques de l’IHÉS*, 53:53–78, 1981.
- 16 Lov K Grover. A fast quantum mechanical algorithm for database search. *Proceedings of the Twenty-Eighth Annual ACM Symposium of Theory of Computing*, pages 212–219, 1996.
- 17 Aram W Harrow, Avinandan Hassidim, and Seth Lloyd. Quantum algorithm for linear systems of equations. *Physical review letters*, 103(15):150502, 2009.
- 18 Fred C Hennie. One-tape, off-line Turing machine computations. *Information and Control*, 8(6):553–578, 1965.



- 19 Thomas Herbst. On a subclass of context-free groups. *RAIRO-Theoretical Informatics and Applications-Informatique Théorique et Applications*, 25(3):255–272, 1991.
- 20 Derek F Holt, Sarah Rees, Claas E Röver, and Richard M Thomas. Groups with context-free co-word problem. *Journal of the London Mathematical Society*, 71(3):643–657, 2005.
- 21 Clara Löh. *Geometric group theory*. Springer, 2017.
- 22 Dieter van Melkebeek and Thomas Watson. Time-space efficient simulations of quantum computations. *Theory of Computing*, 8(1):1–51, 2012.
- 23 David E Muller and Paul E Schupp. Groups, the theory of ends, and context-free languages. *Journal of Computer and System Sciences*, 26(3):295–310, 1983.
- 24 Michael A Nielsen and Isaac Chuang. Quantum computation and quantum information, 2002.
- 25 Michael O Rabin and Dana Scott. Finite automata and their decision problems. *IBM journal of research and development*, 3(2):114–125, 1959.
- 26 Zachary Remscrim. Lower bounds on the running time of two-way quantum finite automata and sublogarithmic-space quantum turing machines, 2020. [arXiv:2003.09877](#).
- 27 Zachary Remscrim. The Power of a Single Qubit: Two-Way Quantum Finite Automata and the Word Problem. In *47th International Colloquium on Automata, Languages, and Programming (ICALP 2020)*, volume 168 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 139:1–139:18, 2020.
- 28 AC Say and Abuzer Yakaryilmaz. Magic coins are useful for small-space quantum machines. *Quantum Information & Computation*, 17(11-12):1027–1043, 2017.
- 29 Jeffrey Shallit. Automaticity IV: sequences, sets, and diversity. *Journal de théorie des nombres de Bordeaux*, 8(2):347–367, 1996.
- 30 Jeffrey Shallit and Yuri Breitbart. Automaticity I: Properties of a measure of descriptonal complexity. *Journal of Computer and System Sciences*, 53(1):10–25, 1996.
- 31 Peter W Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *Proceedings 35th annual symposium on foundations of computer science*. Ieee, 1994.
- 32 Amnon Ta-Shma. Inverting well conditioned matrices in quantum logspace. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 881–890, 2013.
- 33 Andreas Thom. Convergent sequences in discrete groups. *Canadian Mathematical Bulletin*, 56(2):424–433, 2013.
- 34 Jacques Tits. Free subgroups in linear groups. *Journal of Algebra*, 20(2):250–270, 1972.
- 35 John Watrous. On the complexity of simulating space-bounded quantum computations. *Computational Complexity*, 12(1-2):48–84, 2003.
- 36 John Watrous. Encyclopedia of complexity and system science, chapter quantum computational complexity, 2009. [arXiv:0804.3401](#).
- 37 John Watrous. *The theory of quantum information*. Cambridge University Press, 2018.
- 38 Joseph A Wolf et al. Growth of finitely generated solvable groups and curvature of riemannian manifolds. *Journal of differential Geometry*, 2(4):421–446, 1968.
- 39 Abuzer Yakaryilmaz and AC Cem Say. Succinctness of two-way probabilistic and quantum finite automata. *Discrete Mathematics and Theoretical Computer Science*, 12(4):19–40, 2010.

# On the Complexity of $\#CSP^d$

Jiabao Lin

Shanghai University of Finance and Economics, China

jiabaolincs@gmail.com

---

## Abstract

Counting  $CSP^d$  is the counting constraint satisfaction problem ( $\#CSP$  in short) restricted to the instances where every variable occurs a multiple of  $d$  times. This paper revisits tractable structures in  $\#CSP$  and gives a complexity classification theorem for  $\#CSP^d$  with algebraic complex weights. The result unifies affine functions (stabilizer states in quantum information theory) and related variants such as the local affine functions, the discovery of which leads to all the recent progress on the complexity of Holant problems.

The Holant is a framework that generalizes counting CSP. In the literature on Holant problems, weighted constraints are often expressed as tensors (vectors) such that projections and linear transformations help analyze the structure. This paper gives an example showing that different classes of tensors distinguished by these algebraic operations may share the same closure property under tensor product and contraction.

**2012 ACM Subject Classification** Theory of computation → Problems, reductions and completeness

**Keywords and phrases** Constraint satisfaction problem, counting problems, Holant, complexity dichotomy

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.40

**Funding** *Jiabao Lin*: This work is supported by Science and Technology Innovation 2030 –“New Generation of Artificial Intelligence” Major Project No.(2018AAA0100903), NSFC grant 61932002, Program for Innovative Research Team of Shanghai University of Finance and Economics (IRTSHUFE) and the Fundamental Research Funds for the Central Universities.

**Acknowledgements** I would like to thank anonymous referees for their insightful comments and suggestions.

## 1 Introduction

In the constraint satisfaction problem (CSP), constraints are specified by relations on a finite domain  $D = \{0, 1, \dots, q-1\}$  with  $q \geq 2$ . A relation  $R \subseteq D^n$  can be seen as a function  $f_R : D^n \rightarrow \{0, 1\}$  where  $f_R(\mathbf{x}) = 1$  if and only if  $\mathbf{x} \in R$ . To express weighted constraints, we replace relations with complex-valued functions. Let  $\mathbb{C}$  denote the set of algebraic complex numbers. Throughout this paper, we refer to them simply as complex numbers. Let  $\mathcal{F} = \{f_1, \dots, f_l\}$  be a finite function set where  $f_i : D^{n_i} \rightarrow \mathbb{C}$  with arity  $n_i > 0$ . Then the weighted counting CSP specified by the set  $\mathcal{F}$ , denoted by  $\#CSP(\mathcal{F})$ , is defined as follows. An input instance  $I$  of the problem consists of

- A finite set of variables  $V = \{x_1, \dots, x_n\}$ ;
- A finite set of constraints  $\{(F_1, \mathbf{x}_1), \dots, (F_m, \mathbf{x}_m)\}$  where  $F_i \in \mathcal{F}$  and  $\mathbf{x}_i \in V^{\text{arity}(F_i)}$  is a tuple of (not necessarily distinct) variables.

Following [4], we say that the instance  $I$  defines a function of arity  $n$ :

$$F_I(x_1, \dots, x_n) = \prod_{i=1}^m F_i(\mathbf{x}_i).$$



© Jiabao Lin;

licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 40; pp. 40:1–40:10

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



## 40:2 On the Complexity of $\#\text{CSP}^d$

The output of the problem  $\#\text{CSP}(\mathcal{F})$  is the following sum (also called the *partition function*):

$$Z(I) = \sum_{\mathbf{x} \in D^n} F_I(\mathbf{x}).$$

The problem is the counting version of a classical CSP, if we restrict  $\mathcal{F}$  to functions with range  $\{0, 1\}$ . Weighted constraints make the  $\#\text{CSP}$  framework more expressive. For a binary function  $f : D^2 \rightarrow \mathbb{C}$ ,  $\#\text{CSP}(f)$  is the problem of counting graph homomorphisms into the graph on  $D$  with edge weights  $f(i, j)$ . A wide range of graph parameters can be encoded by graph homomorphisms (see, e.g. [22]), which also play an important role in statistical physics.

The complexity of counting CSP has been intensively studied over the last two decades. Bulatov [3] first gave a complexity dichotomy theorem for unweighted  $\#\text{CSP}$ s: Each problem is either solvable in polynomial time or proved to be  $\#\text{P}$ -hard. Understanding the proof requires knowledge of universal algebra. Later, Dyer and Richerby [18] found a new tractability criterion and their proof is elementary. Based on the techniques developed for unweighted  $\#\text{CSP}$ , the dichotomy was generalized to cover nonnegative [6] and complex weights [4]. Given a function  $F : D^n \rightarrow \mathbb{C}$ , we use  $F^{[t]}$ , for each  $t \in [n] = \{1, \dots, n\}$ , to denote the following function of arity  $t$ :

$$F^{[t]}(x_1, \dots, x_t) = \sum_{x_{t+1}, \dots, x_n} F(x_1, \dots, x_t, x_{t+1}, \dots, x_n).$$

And for a function set  $\mathcal{F}$ , we define the set

$$\mathcal{W}_{\mathcal{F}} = \{F^{[t]} \mid F \text{ is a function defined by an instance of } \#\text{CSP}(\mathcal{F}) \text{ and } 1 \leq t \leq \text{arity of } F\}.$$

The dichotomy theorem for complex-weighted  $\#\text{CSP}$  is stated as follows.

► **Theorem 1** ([4]). *Let  $\mathcal{F}$  be a finite set of complex-valued functions. The problem  $\#\text{CSP}(\mathcal{F})$  is solvable in polynomial time if the set  $\mathcal{W}_{\mathcal{F}}$  satisfies three conditions: the Block Orthogonality condition, the Type Partition condition, and the Mal'tsev condition. Otherwise  $\#\text{CSP}(\mathcal{F})$  is  $\#\text{P}$ -hard.*

Roughly speaking, the three conditions require the function defined by any instance to be well-structured, such that the sum of function values can be computed by an efficient algorithm instead of brute-force enumeration. Actually, even if a problem  $\#\text{CSP}(\mathcal{F})$  is  $\#\text{P}$ -hard, it is still possible that the algorithm succeeds on a nontrivial subset of instances. In this paper, we consider a special case denoted by  $\#\text{CSP}^d$ , which was first studied by Huang and Lu [20].

► **Definition 2.** *Let  $d \geq 1$  be an integer and let  $\mathcal{F}$  be a set of complex-valued functions. The problem  $\#\text{CSP}^d(\mathcal{F})$  is the restriction of  $\#\text{CSP}(\mathcal{F})$  to the instances where every variable occurs a multiple of  $d$  times.*

By definition, if  $d = 1$ , then the problem  $\#\text{CSP}^d(\mathcal{F})$  is exactly  $\#\text{CSP}(\mathcal{F})$ . For a function set  $\mathcal{F}$ ,  $\#\text{CSP}^d(\mathcal{F})$  is a subproblem of  $\#\text{CSP}(\mathcal{F})$ . We consider a subset of  $\mathcal{W}_{\mathcal{F}}$ :

$$\mathcal{W}_{\mathcal{F}}^d = \{F^{[t]} \mid F \text{ is a function defined by an instance of } \#\text{CSP}^d(\mathcal{F}) \text{ and } 1 \leq t \leq \text{arity of } F\}.$$

A slight modification of the proof of Theorem 1 yields a unified dichotomy theorem for the  $\#\text{CSP}^d$  family.

► **Theorem 3.** *Let  $d \geq 1$  be an integer and let  $\mathcal{F}$  be a finite set of complex-valued functions. The problem  $\#\text{CSP}^d(\mathcal{F})$  is solvable in polynomial time if the set  $\mathcal{W}_{\mathcal{F}}^d$  satisfies three conditions: the Block Orthogonality condition, the Type Partition condition, and the Mal'tsev condition. Otherwise  $\#\text{CSP}^d(\mathcal{F})$  is  $\#\text{P}$ -hard.*

The tractability criteria survive because, as mentioned before, they are imposed on definable functions and hence not sensitive to how many times a variable appears in an instance. Unfortunately, none of the three conditions is known to be decidable. It is desirable to derive more explicit criteria for constraint functions instead of the functions “generated” by them. However, closed-form formulas or a succinct description of the function values might not exist for arbitrary domains. In graph homomorphisms with real or complex weights [19, 5], the classification of binary functions is explicit but very complicated.

By the definition of  $\#\text{CSP}$ , a variable can occur arbitrarily many times in an instance. However, in many graph satisfaction problems like matchings, vertices are viewed as constraints and edges as variables. That is, each variable appears exactly twice. Inspired by holographic algorithms [24], Cai, Lu, and Xia [13] proposed the Holant framework.

► **Definition 4.** *The problem  $\text{Holant}(\mathcal{F})$  is the restriction of  $\#\text{CSP}(\mathcal{F})$  to the instances where every variable occurs exactly twice.*

On the one hand, the Holant is more expressive than  $\#\text{CSP}$  because any  $\#\text{CSP}(\mathcal{F})$  is polynomial-time equivalent to the problem  $\text{Holant}(\mathcal{F} \cup \{\text{EQ}\})$  where  $\text{EQ}(x_1, x_2, x_3) = 1$  if  $x_1 = x_2 = x_3$  and otherwise  $\text{EQ}(x_1, x_2, x_3) = 0$ . On the other hand, by definition,  $\text{Holant}(\mathcal{F})$  is a subproblem of  $\#\text{CSP}(\mathcal{F})$ . The partition function of a bipartite Holant instance is invariant under the operations of the linear group  $GL_q(\mathbb{C})$  on constraint functions. These operations are also called *holographic transformations* [24, 13], which turn out to be one of the new sources of tractability [10, 21, 2, 23].

Early study of Holant problems has a similar flavor to that of  $\#\text{CSP}$  (see, e.g. [17, 13, 12]). Based on the dichotomy for a special family of Holant problems, Cai, Lu, and Xia [15] gave an explicit criterion for complex-weighted  $\#\text{CSP}$  on the Boolean domain  $\{0, 1\}$  (Boolean  $\#\text{CSP}$  in short).

► **Theorem 5** ([15]). *Let  $\mathcal{F}$  be a set of complex-valued functions on the Boolean domain. Then the problem  $\#\text{CSP}(\mathcal{F})$  is solvable in polynomial time if  $\mathcal{F} \subseteq \mathcal{P}$  or  $\mathcal{F} \subseteq \mathcal{A}$ . Otherwise  $\#\text{CSP}(\mathcal{F})$  is  $\#\text{P}$ -hard.*

The definitions of product-type functions  $\mathcal{P}$  and affine functions  $\mathcal{A}$  are given in Section 3. Affine functions are discovered in the first paper on Holant problems [13], and recently these functions are shown to be equivalent to Clifford gates and stabilizer circuits in quantum computing [11, 1].

Significant progress has been made towards a complexity dichotomy for complex-weighted Holant [7, 21, 2, 23, 14, 9]. However, it remains open even on the Boolean domain. Since it was first proposed, the  $\#\text{CSP}^d$  family becomes increasingly important in understanding the relationship between Holant and  $\#\text{CSP}$ . Cai, Lu, and Xia [16] proved a dichotomy theorem for Boolean  $\#\text{CSP}^2$  and discovered a new tractable class called *local affine* functions (see Definition 15). This result laid the foundation for all the recent progress on real or complex Holant [2, 8, 23]. In fact, several major Holant dichotomies inevitably go through the  $\#\text{CSP}^d$  family whose complexity, before Theorem 3, is known only for some special cases [20, 16, 8]. Moreover, the proofs of these results on  $\#\text{CSP}^d$  are complicated, partially because explicit criteria are expected as in Theorem 5.

Although Theorem 3 shows a complexity dichotomy, it is too general to say much about the Boolean domain. Of course we can check the three conditions and derive a simplified version, but there is a direct generalization of Theorem 5. Both product-type and affine functions have a nice closure property:  $\mathcal{W}_{\mathcal{P}} \subseteq \mathcal{P}$  and  $\mathcal{W}_{\mathcal{A}} \subseteq \mathcal{A}$ . This fact and inspiration from Theorem 3 lead to the following theorem.

► **Theorem 6.** *Let  $d \geq 1$  be an integer and let  $\mathcal{F}$  be a set of complex-valued functions on the Boolean domain. Then the problem  $\#CSP^d(\mathcal{F})$  is solvable in polynomial time if  $\mathcal{W}_{\mathcal{F}}^d \subseteq \mathcal{P}$  or  $\mathcal{W}_{\mathcal{F}}^d \subseteq \mathcal{A}$ . Otherwise  $\#CSP^d(\mathcal{F})$  is  $\#P$ -hard.*

The remainder of this paper is organized as follows. In Section 2, we give a proof of Theorem 3. In Section 3, a simple proof of Theorem 6 is presented. This theorem looks very different from the previous results. For example, holographic transformations seem necessary for the definition of the local affine functions and the algorithm that efficiently solves the problem they define. It will be clear why these transformations disappear in Theorem 6. Some concluding remarks appear in Section 4.

## 2 On General Finite Domains

This section is devoted to the proof Theorem 3. Most of the work was done in [4] and we only show necessary modifications.

Throughout this section, we assume that functions and relations are defined on a fixed finite domain  $D = \{0, 1, \dots, q-1\}$ . And we use  $\mathcal{F}$  to denote a finite set of functions on  $D$ . Let  $\leq_T$  denote the polynomial-time Turing reductions.

► **Lemma 7.** *For any finite set  $\mathcal{G} \subset \mathcal{W}_{\mathcal{F}}^d$ ,  $\#CSP(\mathcal{G}) \leq_T \#CSP^d(\mathcal{F})$ .*

**Proof.** For any function  $f \in \mathcal{G} \subset \mathcal{W}_{\mathcal{F}}^d$ , there exists some instance  $I_f$  of the problem  $\#CSP^d(\mathcal{F})$  such that  $f$  is exactly the function defined by  $I_f$ . Note that  $I_f$  is of constant size, since  $\mathcal{G}$  is a finite set.

Let  $I_1$  be an instance of the problem  $\#CSP(\mathcal{G})$ . We replace each constraint  $(f, \mathbf{x}) \in I_1$  with the instance  $I_f$  (in variables  $\mathbf{x}$ ). Then we get a new instance  $I_2$  of the problem  $\#CSP^d(\mathcal{F})$  because the sum of multiples of an integer  $d$  is still a multiple of  $d$ . It is easy to verify that  $Z(I_1) = Z(I_2)$ . The size of  $I_2$  is polynomial in that of  $I_1$ . ◀

The readers can find the statements of the conditions of Theorem 1 in [4, Subsection 3.1]. We say that the function set  $\mathcal{W}_{\mathcal{F}}^d$  violates any of the three conditions, if there exists a finite set  $\mathcal{G} \subset \mathcal{W}_{\mathcal{F}}^d$  violates any of the three. The hardness part of Theorem 1 can be summarized as follows.

► **Lemma 8** (Lemmas 3.2, 3.4, 3.5 in [4]). *If a finite function set  $\mathcal{G}$  violates one of the three conditions in Theorem 1, then the problem  $\#CSP(\mathcal{G})$  is  $\#P$ -hard.*

**Hardness Part of Theorem 3.** Suppose that a finite set  $\mathcal{G} \subset \mathcal{W}_{\mathcal{F}}^d$  violates any of the three conditions. Then the problem  $\#CSP(\mathcal{G})$  is  $\#P$ -hard by Lemma 8. Moreover,  $\#CSP^d(\mathcal{F})$  is  $\#P$ -hard because  $\#CSP(\mathcal{G}) \leq_T \#CSP^d(\mathcal{F})$ . ◀

Now we consider the algorithmic part.

Consider a relation  $R \subseteq D^n$  and a map  $\varphi : D^3 \rightarrow D$ . We say the relation  $R$  is *closed* under the map  $\varphi$ , if for any three tuples  $\mathbf{u} = (u_i), \mathbf{v} = (v_i), \mathbf{w} = (w_i) \in R$ , it holds that

$$(\varphi(u_1, v_1, w_1), \varphi(u_2, v_2, w_2), \dots, \varphi(u_n, v_n, w_n)) \in R.$$

In this case, we also say  $\varphi$  is a *polymorphism* of the relation  $R$ .

► **Definition 9** (Mal'tsev Polymorphism). *Suppose that a relation  $R \subseteq D^n$  is closed under a map  $\varphi : D^3 \rightarrow D$ . We say that  $\varphi$  is a Mal'tsev polymorphism of  $R$ , if*

$$\varphi(a, b, b) = \varphi(b, b, a) = a$$

for all  $a, b \in D$ .

Given a function  $f$ , we use  $R_f$  denote the relation  $\{\mathbf{x} \in D^n \mid f(\mathbf{x}) \neq 0\}$  (called the *support* of  $f$ ). And for every function  $f \in \mathcal{W}_{\mathcal{F}}^d$  of arity  $n \geq 2$ , a relation  $\Omega_f \subseteq D^{2n-2}$  is defined as:

$$(\mathbf{x}, \mathbf{y}) \in \Omega_f \iff f(\mathbf{x}, *) \text{ and } f(\mathbf{y}, *) \text{ are both nonzero and linearly dependent,}$$

where  $f(\mathbf{x}, *)$  denotes the  $q$ -dimensional vector  $(f(\mathbf{x}, 0), f(\mathbf{x}, 1), \dots, f(\mathbf{x}, q-1))$ . Now we define the set

$$\Lambda_{\mathcal{F}}^d = \{R_f \mid f \in \mathcal{W}_{\mathcal{F}}^d\} \cup \{\Omega_f \mid f \in \mathcal{W}_{\mathcal{F}}^d \text{ of arity } \geq 2\}.$$

► **Definition 10** (The Mal'tsev Condition). *All the relations in  $\Lambda_{\mathcal{F}}^d$  have a common Mal'tsev polymorphism.*

We have the following lemma after checking the algorithm (say, denoted by  $\mathcal{A}$ ) for Theorem 1.

► **Lemma 11.** *Suppose that the set  $\mathcal{W}_{\mathcal{F}}^d$  satisfies the three conditions. Then the algorithm  $\mathcal{A}$  can solve the problem  $\#\text{CSP}^d(\mathcal{F})$  in polynomial time if all the relations in  $\Lambda_{\mathcal{F}}^d \cup \{R_f \mid f \in \mathcal{F}\}$  have a common Mal'tsev polymorphism.*

In fact, the Mal'tsev condition already implies the condition in Lemma 11. This completes the algorithmic part of Theorem 3.

► **Lemma 12.** *Suppose that the Mal'tsev condition holds. Then all the relations in  $\Lambda_{\mathcal{F}}^d \cup \{R_f \mid f \in \mathcal{F}\}$  have a common Mal'tsev polymorphism.*

**Proof.** The conclusion is trivial if  $d = 1$ , since  $\mathcal{F} \subset \mathcal{W}_{\mathcal{F}}$ .

For any function  $f : D^n \rightarrow \mathbb{C}$ , we consider the function  $f^d(\mathbf{x}) = (f(\mathbf{x}))^d$  for all  $\mathbf{x} \in D^n$ . The two functions  $f$  and  $f^d$  have the same support:  $R_f = R_{f^d}$ . Then the conclusion follows because  $f^d \in \mathcal{W}_{\mathcal{F}}^d$  for every function  $f \in \mathcal{F}$ . ◀

Some remarks that may help the readers. A relation  $R \subseteq D^n$  with a Mal'tsev polymorphism can be of exponential size in  $n$ . However, Dyer and Richerby [18] showed that, there is a succinct representation of  $R$  determined by the Mal'tsev polymorphism, called the *witness function*, which has linear size in  $n$ . Here we do not introduce the definition of the witness function. Given an instance  $I$ , the algorithm  $\mathcal{A}$  starts with a witness function of the support  $R_{F_I}$ . The algorithm for constructing witness functions, by Dyer and Richerby, works no matter how many times a variable occurs but only requires a Mal'tsev polymorphism shared by all the relations  $R_f$  for  $f \in \mathcal{F}$ . Lemma 11 covers the requirement, which is satisfied trivially when  $d = 1$  since  $\{R_f \mid f \in \mathcal{F}\} \subset \Lambda_{\mathcal{F}}^1$ . Later, the instance  $I$  is only used for evaluating the function  $F_I$  at some points in  $D^n$ . To compute the sum  $Z(I) = \sum_{\mathbf{x} \in D^n} F_I(\mathbf{x})$ , the algorithm  $\mathcal{A}$  produces a data structure, called the *row representation*, for each  $F_I^{[t]}$  ( $t \in [n]$ ). Now suppose that  $I$  is a  $\#\text{CSP}^d$  instance. By definition,  $F_I^{[t]} \in \mathcal{W}_{\mathcal{F}}^d$  for all  $t \in [n]$ . To obtain the row representations, it is sufficient to impose the three conditions on  $\mathcal{W}_{\mathcal{F}}^d$  under which all the functions and relations involved in the computation are well-structured.

### 3 On the Boolean Domain

In this section, all the functions and relations are defined on the Boolean domain  $D = \{0, 1\}$ .

We start with the definition of product-type functions and affine functions. Let  $EQ(x, y)$  denote the equality function:  $EQ(x, y) = 1$  if  $x = y$ , otherwise  $EQ(x, y) = 0$ . And let  $NE(x, y)$  denote the disequality function:  $NE(x, y) = 1 - EQ(x, y)$ .

► **Definition 13** (Product-Type Functions). *A function is of product type if it is defined by a  $\#CSP$  instance where every constraint function is a unary function or the binary function  $EQ$  or  $NE$ . Let  $\mathcal{P}$  denote the set of all product-type functions.*

A Boolean relation is *affine* if it is the set of solutions to a system of linear equations over the field  $\mathbb{Z}_2$ . We say that  $f$  has affine support if its support is affine. Recall that the support of  $f$  is the relation  $R_f = \{\mathbf{x} \in \{0, 1\}^n \mid f(\mathbf{x}) \neq 0\}$ .

► **Definition 14** (Affine Functions). *A function  $f$  of arity  $n$  is affine if its support is affine and there is a constant  $\lambda \in \mathbb{C}$  such that for all  $\mathbf{x} \in R_f$ ,*

$$f(\mathbf{x}) = \lambda \cdot i^{Q(\mathbf{x})},$$

where  $i = \sqrt{-1}$  and  $Q$  is a homogeneous quadratic polynomial

$$Q(x_1, \dots, x_n) = \sum_{i=1}^n a_i x_i^2 + 2 \sum_{1 \leq i < j \leq n} b_{ij} x_i x_j$$

with  $a_i \in \mathbb{Z}_4$  and  $b_{ij} \in \{0, 1\}$ . We use  $\mathcal{A}$  to denote the set of all affine functions.

**Proof of Theorem 6.** Suppose that there are two functions  $f, g \in \mathcal{W}_{\mathcal{F}}^d$  (they can be the same) such that  $f \notin \mathcal{P}$  and  $g \notin \mathcal{A}$ . By Theorem 5, the problem  $\#CSP(\{f, g\})$  is  $\#P$ -hard. Then  $\#CSP^d(\mathcal{F})$  is also  $\#P$ -hard since  $\#CSP(\{f, g\}) \leq_{\tau} \#CSP^d(\mathcal{F})$  by Lemma 7.

Now we assume that  $\mathcal{W}_{\mathcal{F}}^d \subseteq \mathcal{P}$  or  $\mathcal{W}_{\mathcal{F}}^d \subseteq \mathcal{A}$ . In both cases, for any instance  $I$ , the function  $F_I$  (say, of arity  $n$ ) has affine support and the linear system for the support can be constructed efficiently from that of the constraint functions. There are two cases:

- $F_I \in \mathcal{P}$ . We can determine the variable dependence on the support:  $x_i = x_j$  or  $x_i \neq x_j$  or they are independent. Then the evaluation of the partition function  $Z(I)$  reduces to a trivial case where every constraint is unary.
- $F_I \in \mathcal{A}$ . We can obtain the explicit formula (in Definition 14) for the function  $F_I$ , by evaluating it at  $O(n^2)$  many points, using the instance  $I$ . Then the algorithm for Theorem 5 is able to compute  $Z(I)$ . See [15] for more details.

Therefore, the partition function is computable in polynomial time. ◀

The remainder of this section is devoted to the connection between Theorem 6 and the dichotomy for Boolean  $\#CSP^2$  in [16]. Before this, we need to introduce some notations and definitions.

A function of arity  $n \geq 2$  can be expressed as a  $2^{n-r} \times 2^r$  matrix ( $0 \leq r \leq n$ ), denoted by  $M_r(f)$ . The rows and columns are indexed by  $\mathbf{x} \in \{0, 1\}^{n-r}$  and  $\mathbf{y} \in \{0, 1\}^r$ , respectively, and  $f(\mathbf{x}, \mathbf{y})$  is the  $(\mathbf{x}, \mathbf{y})^{\text{th}}$  entry of the matrix  $M_r(f)$ . In particular, when  $r = 0$ ,  $M_r(f)$  is a column vector of dimension  $2^n$ . In the following, we do not distinguish a function from its matrix representations. The integer  $r$  will be clear in matrix multiplication.

Given two matrices  $A$  and  $B$ , we use  $A \otimes B$  to denote their tensor product.

Let  $\alpha = \frac{1+i}{\sqrt{2}}$  where  $i = \sqrt{-1}$ . Then  $\alpha^2 = i$ . And let  $M_x = \begin{bmatrix} 1 & 0 \\ 0 & x \end{bmatrix}$  for  $x \in \mathbb{C}$ . We define the following set of functions:

$$\mathcal{A}^\alpha = \{M_\alpha^{\otimes n} g \mid g \in \mathcal{A} \text{ and } n \text{ is the arity of } g\},$$

which is a tractable class for  $\#\text{CSP}^2$ .

Recall that  $R_f = \{\mathbf{x} \in \{0, 1\}^n \mid f(\mathbf{x}) \neq 0\}$  for a function  $f$  of arity  $n$ .

► **Definition 15** (Local Affine Functions). *A function  $f$  (of arity  $n$ ) is called local affine, if for every element  $(s_1, s_2, \dots, s_n) \in R_f$ ,*

$$(M_{\alpha^{s_1}} \otimes M_{\alpha^{s_2}} \otimes \dots \otimes M_{\alpha^{s_n}})f \in \mathcal{A},$$

where  $\alpha^s = 1$  if  $s = 0$ ,  $\alpha^s = \alpha$  if  $s = 1$ . The set of all local affine functions is denoted by  $\mathcal{L}$ .

► **Theorem 16** ([16]). *Let  $\mathcal{F}$  be a set of functions on the Boolean domain. If  $\mathcal{F} \subseteq \mathcal{C}$  for  $\mathcal{C} \in \{\mathcal{P}, \mathcal{A}, \mathcal{A}^\alpha, \mathcal{L}\}$ , then the problem  $\#\text{CSP}^2(\mathcal{F})$  is solvable in polynomial time. Otherwise  $\#\text{CSP}^2(\mathcal{F})$  is  $\#\text{P}$ -hard.*

Since the theorem above is a special case of Theorem 6, the two tractability criteria should be compatible. In fact, we have the following observation.

► **Lemma 17.** *If  $\mathcal{F} \subseteq \mathcal{C}$  for  $\mathcal{C} \in \{\mathcal{P}, \mathcal{A}, \mathcal{A}^\alpha, \mathcal{L}\}$ , then  $\mathcal{W}_{\mathcal{F}}^2 \subseteq \mathcal{P}$  or  $\mathcal{W}_{\mathcal{F}}^2 \subseteq \mathcal{A}$ .*

The algorithms in [16] for the two classes  $\mathcal{A}^\alpha$  and  $\mathcal{L}$  start with local transformations induced by the matrix  $M_\alpha$ , such that every constraint function of an instance  $I$  becomes affine and hence the function  $F_I$  is also affine. However, as stated in Lemma 17,  $F_I$  is already affine. Before proving the lemma, we need some preparations.

► **Lemma 18** (Closure Property). *Let  $\mathcal{C} = \mathcal{P}$  or  $\mathcal{C} = \mathcal{A}$ , then  $\mathcal{W}_{\mathcal{F}} \subseteq \mathcal{C}$  for any set  $\mathcal{F} \subseteq \mathcal{C}$ . In particular, for any  $n$ -ary function  $f \in \mathcal{C}$ :*

- $f^{[t]} \in \mathcal{C}$  for all  $t \in [n]$ ;
- $f_\pi \in \mathcal{C}$  for any permutation  $\pi$  on  $[n]$ , where  $f_\pi(x_1, x_2, \dots, x_n) = f(x_{\pi(1)}, x_{\pi(2)}, \dots, x_{\pi(n)})$ .

**Proof.** We only show that  $g(x_1, \dots, x_{n-1}) = \sum_{x_n \in \{0,1\}} f(x_1, \dots, x_{n-1}, x_n) \in \mathcal{C}$  for any  $n$ -ary function  $f \in \mathcal{C}$ . The case  $\mathcal{C} = \mathcal{A}$  was proved in [11, Lemma 3.1]. Now suppose that  $f \in \mathcal{P}$ . By the definition of product-type functions, it is sufficient to consider the case where  $f$  has support

$$R_f \subseteq \{(u_1, \dots, u_n), (1 - u_1, \dots, 1 - u_n)\} \text{ for some } u_i \in \{0, 1\}.$$

It then follows that  $R_g \subseteq \{(u_1, \dots, u_{n-1}), (1 - u_1, \dots, 1 - u_{n-1})\}$ . Thus  $g \in \mathcal{P}$ . ◀

► **Lemma 19** (Closure under Matrix Multiplication). *Let  $f \in \mathcal{A}$  be a function of arity  $n$ . And let  $g \in \mathcal{A}$  be a function of arity  $m$ . Then it holds that*

$$M_r(f)M_{m-r}(g) \in \mathcal{A},$$

for all  $0 \leq r \leq \min\{n, m\}$ .

**Proof.** For any  $0 \leq r \leq \min\{n, m\}$ , we consider the function of arity  $n + m - 2r$ :

$$\begin{aligned} & h(x_1, \dots, x_{n-r}, y_1, \dots, y_{m-r}) \\ &= \sum_{z_1, \dots, z_r \in \{0,1\}} f(x_1, \dots, x_{n-r}, z_1, \dots, z_r) g(z_1, \dots, z_r, y_1, \dots, y_{m-r}). \end{aligned}$$

Then it follows that  $M_{m-r}(h) = M_r(f)M_{m-r}(g)$ . By Lemma 18, we have  $h \in \mathcal{A}$ . ◀

► **Lemma 20** (Closure under Tensor Product). *Let  $A, B \in \mathcal{A}$  be two matrices. Then  $A \otimes B \in \mathcal{A}$ .*

**Proof.** Let  $f, g \in \mathcal{A}$  be two functions such that  $A = M_r(f)$  and  $B = M_t(g)$  for some integers  $r, t \geq 0$ . Consider the function

$$h(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{w}) = f(\mathbf{x}, \mathbf{z})g(\mathbf{y}, \mathbf{w})$$

where  $\mathbf{z} \in \{0, 1\}^r$  and  $\mathbf{w} \in \{0, 1\}^t$ . Then  $A \otimes B = M_{r+t}(h)$ . ◀

► **Lemma 21.** *Let  $f$  be a function of arity  $n$ . Suppose that there exist matrices  $A_1, A_2, \dots, A_n \in \left\{ \begin{bmatrix} 1 & 0 \\ 0 & i^r \end{bmatrix} \mid r = 0, 1, 2, 3 \right\}$  where  $i = \sqrt{-1}$ , such that  $(A_1 \otimes A_2 \otimes \dots \otimes A_n)f \in \mathcal{A}$ . Then  $f \in \mathcal{A}$ .*

**Proof.** Set  $A = A_1 \otimes A_2 \otimes \dots \otimes A_n$ . The matrix  $A$  is invertible and  $A^{-1} = A_1^{-1} \otimes A_2^{-1} \otimes \dots \otimes A_n^{-1} \in \mathcal{A}$  by Lemma 20. Then it follows that  $f = (A^{-1}A)f = A^{-1}(Af) \in \mathcal{A}$ , according to Lemma 19. ◀

Now we are ready to prove Lemma 17.

**Proof of Lemma 17.** Due to the closure property of product-type and affine functions (Lemma 18), the two cases  $\mathcal{F} \subseteq \mathcal{A}^\alpha$  and  $\mathcal{F} \subseteq \mathcal{L}$  remain to be verified. Furthermore, by Lemma 18, we only need to check the definable functions.

Let  $I$  be an instance of  $\#CSP^2(\mathcal{F})$ . Suppose that the instance  $I$  has  $n$  variables  $\{x_1, x_2, \dots, x_n\}$  and  $m$  constraints  $\{(f_1, \mathbf{x}_1), (f_2, \mathbf{x}_2), \dots, (f_m, \mathbf{x}_m)\}$ . Then the function defined by  $I$  is

$$F_I(x_1, \dots, x_n) = f_1(\mathbf{x}_1)f_2(\mathbf{x}_2) \cdots f_m(\mathbf{x}_m).$$

Suppose that each variable  $x_j$  occurs  $k_j$  times in the instance  $I$ . By definition,  $k_j$  is even for each  $j \in [n]$  and we set  $k = k_1 + k_2 + \dots + k_n$ .

Consider the function of arity  $k$ :  $g = f_1 \otimes f_2 \otimes \dots \otimes f_m$ . There is a permutation  $\pi$  on  $[k]$  such that

$$F_I(x_1, \dots, x_n) = g_\pi(\underbrace{x_1, \dots, x_1}_{k_1 \text{ times}}, \underbrace{x_2, \dots, x_2}_{k_2 \text{ times}}, \dots, \underbrace{x_n, \dots, x_n}_{k_n \text{ times}}).$$

Suppose that  $\mathcal{F} \subseteq \mathcal{L}$ . We show that  $F_I \in \mathcal{A}$ . If the function  $F_I$  is identically zero, then we are done. Suppose not. Then there exist  $s_1, s_2, \dots, s_n \in \{0, 1\}$  such that

$$F_I(s_1, s_2, \dots, s_n) = g_\pi(\underbrace{s_1, \dots, s_1}_{k_1 \text{ times}}, \underbrace{s_2, \dots, s_2}_{k_2 \text{ times}}, \dots, \underbrace{s_n, \dots, s_n}_{k_n \text{ times}}) \neq 0.$$

By the definition of local affine functions, we have

$$(M_{\alpha^{s_1}}^{\otimes k_1} \otimes M_{\alpha^{s_2}}^{\otimes k_2} \otimes \dots \otimes M_{\alpha^{s_n}}^{\otimes k_n})g_\pi \in \mathcal{A}. \quad (*)$$

The relation between the two functions  $F_I$  and  $g_\pi$  shows that

$$(M_{\alpha^{s_1}}^{k_1} \otimes M_{\alpha^{s_2}}^{k_2} \otimes \dots \otimes M_{\alpha^{s_n}}^{k_n})F_I \in \mathcal{A}.$$

For each  $j \in [n]$ ,  $M_{\alpha^{s_j}}^{k_j} \in \left\{ \begin{bmatrix} 1 & 0 \\ 0 & i^r \end{bmatrix} \mid r = 0, 1, 2, 3 \right\}$ , since  $k_j$  is even. By Lemma 21, we have  $F_I \in \mathcal{A}$ .

Now suppose that  $\mathcal{F} \subseteq \mathcal{A}^\alpha$ . However, this case has been considered, because the relation  $(*)$  holds by setting  $s_1 = s_2 = \dots = s_n = 1$ . This completes the proof. ◀



## 4 Conclusion

Local affine functions partially reflect the difficulty in proving a Holant dichotomy: Nice structures hide in strange supports and we lack powerful tools.

Theorem 3 and Theorem 6 give a unified complexity dichotomy for the whole  $\#\text{CSP}^d$  family. Being abstract enough, they reveal that essentially there is no new tractable structure for  $d > 1$ . This fact is obtained by considering barriers to efficient evaluations of the partition functions, but not simply the set of constraint functions which defines the problem though. Moreover, the proof of Theorem 6 is much simpler than those of the partial results on  $\#\text{CSP}^d$ .

It is not clear whether or not the dichotomies in this paper can help the study of Holant problems at large. They are much more conceptual than existing dichotomies and techniques for the Holant.

---

## References

- 1 Miriam Backens. A new holant dichotomy inspired by quantum computation. In *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017*, pages 16:1–16:14, 2017. doi:10.4230/LIPIcs.ICALP.2017.16.
- 2 Miriam Backens. A complete dichotomy for complex-valued Holant<sup>c</sup>. In *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, pages 12:1–12:14, 2018. doi:10.4230/LIPIcs.ICALP.2018.12.
- 3 Andrei A. Bulatov. The complexity of the counting constraint satisfaction problem. *J. ACM*, 60(5):34, 2013. doi:10.1145/2528400.
- 4 Jin-Yi Cai and Xi Chen. Complexity of counting CSP with complex weights. *J. ACM*, 64(3):19:1–19:39, 2017. doi:10.1145/2822891.
- 5 Jin-Yi Cai, Xi Chen, and Pinyan Lu. Graph homomorphisms with complex values: A dichotomy theorem. *SIAM J. Comput.*, 42(3):934–1029, 2013. doi:10.1137/110840194.
- 6 Jin-Yi Cai, Xi Chen, and Pinyan Lu. Nonnegative weighted  $\#\text{CSP}$ : An effective complexity dichotomy. *SIAM J. Comput.*, 45(6):2177–2198, 2016. doi:10.1137/15m1032314.
- 7 Jin-Yi Cai, Zhiguo Fu, Heng Guo, and Tyson Williams. A Holant dichotomy: Is the FKT algorithm universal? In *IEEE 56th Annual Symposium on Foundations of Computer Science*, pages 1259–1276, 2015. doi:10.1109/focs.2015.81.
- 8 Jin-Yi Cai, Zhiguo Fu, and Shuai Shao. From Holant to quantum entanglement and back. In *47th International Colloquium on Automata, Languages, and Programming, ICALP 2020, July 8-11, 2020, Saarbrücken, Germany (Virtual Conference)*, pages 22:1–22:16, 2020. doi:10.4230/LIPIcs.ICALP.2020.22.
- 9 Jin-Yi Cai, Heng Guo, and Tyson Williams. The complexity of counting edge colorings and a dichotomy for some higher domain Holant problems. In *55th IEEE Annual Symposium on Foundations of Computer Science*, pages 601–610, 2014. doi:10.1109/F0CS.2014.70.
- 10 Jin-Yi Cai, Heng Guo, and Tyson Williams. A complete dichotomy rises from the capture of vanishing signatures. *SIAM J. Comput.*, 45(5):1671–1728, 2016. doi:10.1137/15m1049798.
- 11 Jin-Yi Cai, Heng Guo, and Tyson Williams. Clifford gates in the holant framework. *Theor. Comput. Sci.*, 745:163–171, 2018. doi:10.1016/j.tcs.2018.06.010.
- 12 Jin-Yi Cai, Sangxia Huang, and Pinyan Lu. From Holant to  $\#\text{CSP}$  and back: Dichotomy for Holant<sup>c</sup> problems. *Algorithmica*, 64(3):511–533, 2012. doi:10.1007/s00453-012-9626-6.
- 13 Jin-Yi Cai, Pinyan Lu, and Mingji Xia. Holant problems and counting CSP. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing*, pages 715–724, 2009. doi:10.1145/1536414.1536511.
- 14 Jin-Yi Cai, Pinyan Lu, and Mingji Xia. Dichotomy for Holant\* problems with domain size 3. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1278–1295, 2013. doi:10.1137/1.9781611973105.93.

- 15 Jin-Yi Cai, Pinyan Lu, and Mingji Xia. The complexity of complex weighted Boolean  $\#CSP$ . *J. Comput. Syst. Sci.*, 80(1):217–236, 2014. doi:10.1016/j.jcss.2013.07.003.
- 16 Jin-Yi Cai, Pinyan Lu, and Mingji Xia. Dichotomy for real Holant<sup>c</sup> problems. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 1802–1821, 2018. doi:10.1137/1.9781611975031.118.
- 17 Martin E. Dyer, Leslie A. Goldberg, and Mark Jerrum. The complexity of weighted Boolean  $\#CSP$ . *SIAM J. Comput.*, 38(5):1970–1986, 2009. doi:10.1137/070690201.
- 18 Martin E. Dyer and David Richerby. An effective dichotomy for the counting constraint satisfaction problem. *SIAM J. Comput.*, 42(3):1245–1274, 2013. doi:10.1137/100811258.
- 19 Leslie A. Goldberg, Martin Grohe, Mark Jerrum, and Marc Thurley. A complexity dichotomy for partition functions with mixed signs. *SIAM J. Comput.*, 39(7):3336–3402, 2010. doi:10.1137/090757496.
- 20 Sangxia Huang and Pinyan Lu. A dichotomy for real weighted Holant problems. *Computational Complexity*, 25(1):255–304, 2016. doi:10.1007/s00037-015-0118-3.
- 21 Jiabao Lin and Hanpin Wang. The complexity of Boolean Holant problems with nonnegative weights. *SIAM J. Comput.*, 47(3):798–828, 2018. doi:10.1137/17M113304X.
- 22 László Lovász. *Large Networks and Graph Limits*, volume 60 of *Colloquium Publications*. American Mathematical Society, 2012. URL: <http://www.ams.org/bookstore-getitem/item=COLL-60>.
- 23 Shuai Shao and Jin-Yi Cai. A dichotomy for real Boolean Holant problems. *To appear at FOCS*, 2020. arXiv:2005.07906.
- 24 Leslie G. Valiant. Holographic algorithms. *SIAM J. Comput.*, 37(5):1565–1594, 2008. doi:10.1137/070682575.

# Interactive Proofs for Verifying Machine Learning

**Shafi Goldwasser** 

University of California, Berkeley, CA, USA

<https://simons.berkeley.edu/people/shafi-goldwasser>


[shafi.goldwasser@berkeley.edu](mailto:shafi.goldwasser@berkeley.edu)

**Guy N. Rothblum** 

Weizmann Institute of Science, Rehovot, Israel

<https://guyrothblum.wordpress.com>

[rothblum@alum.mit.edu](mailto:rothblum@alum.mit.edu)

**Jonathan Shafer** 

University of California, Berkeley, CA, USA

<https://shaferjo.com>

[shaferjo@berkeley.edu](mailto:shaferjo@berkeley.edu)

**Amir Yehudayoff** 

Technion – Israel Institute of Technology, Haifa, Israel

<https://yehudayoff.net.technion.ac.il>

[amir.yehudayoff@gmail.com](mailto:amir.yehudayoff@gmail.com)

---

## Abstract

We consider the following question: using a source of labeled data and interaction with an untrusted prover, what is the complexity of verifying that a given hypothesis is “approximately correct”? We study interactive proof systems for *PAC verification*, where a verifier that interacts with a prover is required to accept good hypotheses, and reject bad hypotheses. Both the verifier and the prover are efficient and have access to labeled data samples from an unknown distribution. We are interested in cases where the verifier can use significantly less data than is required for (agnostic) PAC learning, or use a substantially cheaper data source (e.g., using only random samples for verification, even though learning requires membership queries). We believe that today, when data and data-driven algorithms are quickly gaining prominence, the question of verifying purported outcomes of data analyses is very well-motivated.

We show three main results. First, we prove that for a specific hypothesis class, verification is significantly cheaper than learning in terms of sample complexity, even if the verifier engages with the prover only in a single-round (NP-like) protocol. Moreover, for this class we prove that single-round verification is also significantly cheaper than testing closeness to the class. Second, for the broad class of Fourier-sparse boolean functions, we show a multi-round (IP-like) verification protocol, where the prover uses membership queries, and the verifier is able to assess the result while only using random samples. Third, we show that verification is not always more efficient. Namely, we show a class of functions where verification requires as many samples as learning does, up to a logarithmic factor.

**2012 ACM Subject Classification** Theory of computation → Interactive proof systems; Theory of computation → Machine learning theory

**Keywords and phrases** PAC learning, Fourier analysis of boolean functions, Complexity gaps, Complexity lower bounds, Goldreich-Levin algorithm, Kushilevitz-Mansour algorithm, Distribution testing

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.41

**Related Version** A full version of the paper is available at <https://eccc.weizmann.ac.il/report/2020/058>.



© Shafi Goldwasser, Guy N. Rothblum, Jonathan Shafer, and Amir Yehudayoff;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 41; pp. 41:1–41:19



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

**Funding** *Shafi Goldwasser*: Research was supported in part by DARPA under Contract No. HR001120C0015.

*Guy N. Rothblum*: This project has received funding from the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation programme (grant agreement No. 819702).

*Amir Yehudayoff*: Research was supported by ISF grant No. 1162/15.

**Acknowledgements** The authors would like to thank Oded Ben-David, Eyal Ben-David, Alessandro Chiesa, Constantinos Daskalakis, Oded Goldreich, Ankur Moitra, Ido Nachum, Orr Paradise and Avishay Tal for insightful discussions and helpful references.

---

*A simple idea underpins science: “trust, but verify”. Results should always be subject to challenge from experiment. That simple but powerful idea has generated a vast body of knowledge. Since its birth in the 17th century, modern science has changed the world beyond recognition, and overwhelmingly for the better. But success can breed complacency. Modern scientists are doing too much trusting and not enough verifying – to the detriment of the whole of science, and of humanity.*

The Economist, “How Science Goes Wrong” (2013)

## 1 Introduction

Data and data-driven algorithms are transforming science and society. State-of-the-art machine learning and statistical analysis algorithms use access to data at scales and granularities that would have been unimaginable even a few years ago. From medical records and genomic information to financial transactions and transportation networks, this revolution spans scientific studies, commercial applications and the operation of governments. It holds transformational promise, but also raises new concerns. If data analysis requires huge amounts of data and computational power, how can one verify the correctness and accuracy of the results? Might there be asymmetric cases, where performing the analysis is expensive, but verification is cheap?

There are many types of statistical analyses, and many ways to formalize the notion of verifying the outcome. In this work we focus on interactive proof systems [12] for verifying supervised learning, as defined by the PAC model of learning [22]. Our emphasis throughout is on access to the underlying data distribution as the critical resource: both quantitatively (how many samples are used for learning versus for verification), and qualitatively (what types of samples are used). We embark on tackling a series of new questions:

Suppose a learner (which we also call *prover*) claims to have arrived at a good hypothesis with regard to an unknown data distribution by analyzing random samples from the distribution. Can one verify the quality of the hypothesis with respect to the unknown distribution by using significantly fewer samples than the number needed to independently repeat the analysis? The crucial difference between this question and questions that appear in the property testing and distribution testing literature is that we allow the prover and verifier to engage in an *interactive* communication protocol (see Section 1.1.1 for a comparison). We are interested in the case where both the verifier and an honest prover are efficient (i.e., use polynomial runtime and sample complexity), and furthermore, a dishonest prover with unbounded computational resources cannot fool the verifier:

► **Question 1 (Runtime and sample complexity of learning vs. verifying).** *Are there machine learning tasks for which the runtime and sample complexity of learning a good hypothesis is significantly larger than the complexity of verifying a hypothesis provided by someone else?*

In the learning theory literature, various types of access to training data have been considered, such as random samples, membership queries, and statistical queries. In the real world, some types of access are more costly than others. Therefore, it is interesting to consider whether it is possible to verify a hypothesis using a cheaper type of access than is necessary for learning:

► **Question 2 (Sample type of learning vs. verifying).** *Are there machine learning problems where membership queries are necessary for finding a good hypothesis, but verification is possible using random samples alone?*

The answers to these fundamental questions are motivated by real-world applications. If data analysis requires huge amounts of data and computational resources while verification is a simpler task, then a natural approach for individuals and weaker entities would be to delegate the data collection and analysis to more powerful entities. Going beyond machine learning, this applies also to verifying the results of scientific studies without replicating the entire experiment. We elaborate on these motivating applications in Section 1.2 below.

## 1.1 PAC Verification: A Proposed Model

Our primary focus in this work is verifying the results of agnostic supervised machine learning algorithms that receive a labeled dataset, and aim to learn a classifier that predicts the labels of unseen examples. We introduce a notion of interactive proof systems for verification of PAC learning, which we call *PAC Verification* (see Definition 4). Here, the entity running the learning algorithms (which we refer to as the *prover* or the *learner*) proves the correctness of the results by engaging in an interactive communication protocol with a verifier. One special case is where the prover only sends a single message constituting an (NP-like) certificate of correctness. The honest prover should be able to convince the verifier to accept its proposed hypothesis with high probability. A dishonest prover (even an unbounded one) should not be able to convince the verifier to accept a hypothesis that is not sufficiently good (as defined below), except with small probability over the verifier's random coins and samples. The proof system is interesting if the amount of resources used for verification is significantly smaller than what is needed for performing the learning task. We are especially interested in *doubly-efficient* proof systems [11], where the honest prover also runs in polynomial time.

More formally, let  $\mathcal{X}$  be a set, and consider a distribution  $\mathcal{D}$  over samples of the form  $(x, y)$  where  $x \in \mathcal{X}$  and  $y \in \{0, 1\}$ . Assume there is some *hypothesis class*  $\mathcal{H}$ , which is a set of functions  $\mathcal{X} \rightarrow \{0, 1\}$ , and we are interested in finding a function  $h \in \mathcal{H}$  that predicts the label  $y$  given a previously unseen  $x$  with high accuracy with respect to  $\mathcal{D}$ . To capture this we use the *loss function*  $L_{\mathcal{D}}(h) = \mathbb{P}_{(x,y) \in \mathcal{D}} [h(x) \neq y]$ . Our goal is to design protocols consisting of a prover and verifier that satisfy: (i) When the verifier interacts with an honest prover, with high probability the verifier outputs a hypothesis  $h$  that is  $\varepsilon$ -good, meaning that

$$L_{\mathcal{D}}(h) \leq L_{\mathcal{D}}(\mathcal{H}) + \varepsilon, \tag{1}$$

where  $L_{\mathcal{D}}(\mathcal{H}) = \inf_{f \in \mathcal{H}} L_{\mathcal{D}}(f)$ ; (ii) For any (possibly dishonest and unbounded) prover, the verifier can choose to reject the interaction, and with high probability the verifier will not output a hypothesis that is not  $\varepsilon$ -good.

Observe that in the *realizable case* (or *promise case*), where we assume that  $L_{\mathcal{D}}(\mathcal{H}) = 0$ , one immediately obtains a strong result: given a hypothesis  $\tilde{h}$  proposed by the prover, a natural strategy for the verifier is to take a few samples from  $\mathcal{D}$ , and accept if and only if  $\tilde{h}$  classifies at most, say, an  $\frac{9}{10}\varepsilon$ -fraction of them incorrectly. From Hoeffding's inequality,

taking  $O(\frac{1}{\varepsilon^2})$  samples is sufficient to ensure that with probability at least  $\frac{9}{10}$  the *empirical loss*<sup>1</sup> of  $\tilde{h}$  is  $\frac{\varepsilon}{10}$ -close to the true loss. Therefore, if  $L_{\mathcal{D}}(\tilde{h}) \leq \frac{8}{10}\varepsilon$  then  $\tilde{h}$  is accepted with probability at least  $\frac{9}{10}$ , and if  $L_{\mathcal{D}}(\tilde{h}) > \varepsilon$  then  $\tilde{h}$  is rejected with probability at least  $\frac{9}{10}$ . In contrast, PAC learning a hypothesis that with probability at least  $\frac{9}{10}$  has loss at most  $\varepsilon$  requires  $\Omega(\frac{d}{\varepsilon})$  samples, where the parameter  $d$ , which is the VC dimension of the class, can be arbitrarily large.<sup>2</sup> That is, in the realizable case there is a sample complexity and time complexity separation of unbounded magnitude between learning and verifying. Furthermore, this result holds also under the weaker assumption that  $L_{\mathcal{D}}(\mathcal{H}) \leq \frac{\varepsilon}{2}$ .

Encouraged by this strong result, the present paper focuses on the *agnostic case*, where no assumptions are made regarding  $L_{\mathcal{D}}(\mathcal{H})$ . Here, things become more interesting, and deciding whether a proposed hypothesis  $\tilde{h}$  is  $\varepsilon$ -good is non-trivial. Indeed, the verifier can efficiently estimate  $L_{\mathcal{D}}(\tilde{h})$  using Hoeffding's inequality as before, but estimating the term  $L_{\mathcal{D}}(\mathcal{H})$  on the right hand side of (1) is considerably more challenging. If  $\tilde{h}$  has a loss of say 15%, it could be an amazingly-good hypothesis compared to the other members of  $\mathcal{H}$ , or it could be very poor. Distinguishing between these two cases may be difficult when  $\mathcal{H}$  is a large and complicated class.

### 1.1.1 Related Models

We discuss two related models studied in prior work, and their relationship to the PAC verification model proposed in this work.

**Property Testing.** Goldreich, Goldwasser and Ron [9] initiated the study of a property testing problem that naturally accompanies proper PAC learning: Given access to samples from an unknown distribution  $\mathcal{D}$ , decide whether  $L_{\mathcal{D}}(\mathcal{H}) = 0$  or  $L_{\mathcal{D}}(\mathcal{H}) \geq \varepsilon$  for some fixed hypothesis class  $\mathcal{H}$ . Further developments and variations appeared in Kearns and Ron [15] and Balcan et. al [2]. Blum and Hu [4] consider *tolerant* closeness testing and a related task of distance approximation (see [19]), where the algorithm is required to approximate  $L_{\mathcal{D}}(\mathcal{H})$  up to a small additive error. As discussed above, the main challenge faced by the verifier in PAC verification is approximating  $L_{\mathcal{D}}(\mathcal{H})$ . However, there is a crucial difference between testing and PAC verification: In addition to taking samples from  $\mathcal{D}$ , the verifier in PAC verification can also interact with a prover, and thus PAC verification can (potentially) be easier than testing. Indeed, this difference is exemplified by the *proper* testing question, where we only need to distinguish the zero-loss case from large loss. As discussed above, proper PAC verification is trivial. Proper testing, on the other hand, can be a challenging goal (and, indeed, has been the focus of a rich body of work). For the *tolerant* setting, we prove a separation between testing and PAC verification: we show a hypothesis class for which the help of the prover allows the verifier to save a (roughly) quadratic factor over the number of samples that are required for closeness testing or distance approximation. See Section 3 for further details.

**Proofs of Proximity for Distributions.** Chiesa and Gur [6] study interactive proof systems for distribution testing. For some fixed property  $\Pi$ , the verifier receives samples from an unknown distribution  $\mathcal{D}$ , and interacts with a prover to decide whether  $\mathcal{D} \in \Pi$  or whether  $\mathcal{D}$  is  $\varepsilon$ -far in total variation distance from any distribution in  $\Pi$ . While that work does not consider

<sup>1</sup> I.e., the fraction of the samples that is misclassified.

<sup>2</sup> See preliminaries in [13, Section 1.6.2] for more about VC dimension.

machine learning, the question of verifying a lower bound  $\ell$  on the loss of a hypothesis class can be viewed as a special case of distribution testing, where  $\Pi = \{\mathcal{D} : L_{\mathcal{D}}(\mathcal{H}) \geq \ell\}$ . Beyond our focus on PAC verification, an important distinction between the works is that in Chiesa and Gur’s model and results, the honest prover’s access to the distribution is unlimited – the honest prover can have complete information about the distribution. In this paper, we focus on doubly-efficient proofs, where the verifier and the honest prover must both be efficient in the number of data samples they require. With real-world applications in mind, this focus seems quite natural.<sup>3</sup>

We survey further related works in Section 1.5 in the full version of the paper [13].

## 1.2 Applications

The P vs. NP problem asks whether finding a solution ourselves is harder than verifying a solution supplied by someone else. It is natural to ask a similar question in learning theory: Are there machine learning problems for which learning a good hypothesis is harder than verifying one proposed by someone else? We find this theoretical motivation compelling in and of itself. Nevertheless, we now proceed to elaborate on a few more practical aspects of this question.

### 1.2.1 Delegation of Learning

In a commercial context, consider a scenario in which a client is interested in developing a machine learning (ML) model, and decides to outsource that task to a company  $P$  that provides ML services. For example,  $P$  promises to train a deep neural net using a big server farm. Furthermore,  $P$  claims to possess a large amount of high quality data that is not available to the client, and promises to use that data for training.

How could the client ascertain that a model provided by  $P$  is actually a good model? The client could use a general-purpose cryptographic delegation-of-computation protocol, but that would be insufficient. Indeed, a general-purpose delegation protocol can only ensure that  $P$  executed the computation as promised, but it cannot provide any guarantees about the quality of the outcome, and in particular cannot ensure that the outcome is  $\varepsilon$ -good: If  $P$  used skewed or otherwise low-quality training data (whether maliciously or inadvertently), a general-purpose delegation protocol has no way of detecting that. Moreover, even if the the data and the execution of the computation were both flawless, this still provides no guarantees on the quality of the output, because an ML model might have poor performance despite being trained as prescribed.<sup>4,5</sup>

A different solution could be to have  $P$  provide a proof to establish that its output is indeed  $\varepsilon$ -good. In cases where the resource gap between learning and verifying is significant enough, the client could cost-effectively verify the proof, obtaining sound guarantees on the quality of the ML model it is purchasing from  $P$ .

<sup>3</sup> In Chiesa and Gur’s setting, it would also be sufficient for the prover to only know the distribution up to  $O(\varepsilon)$  total variation distance, and this can be achieved using random samples from the distribution. However, the number of samples necessary for the prover would be linear in the domain size, which is typically exponential, and so this approach would not work for constructing doubly-efficient PAC verification protocols.

<sup>4</sup> E.g., a neural network might get stuck at a local minimum.

<sup>5</sup> Additionally, note that state-of-the-art delegation protocols are not efficient enough at present to make it practicable to delegate intensive ML computations. See the survey by Walfish and Blumberg [23] for progress and challenges in developing such systems.



### 1.2.2 Verification of Scientific Studies

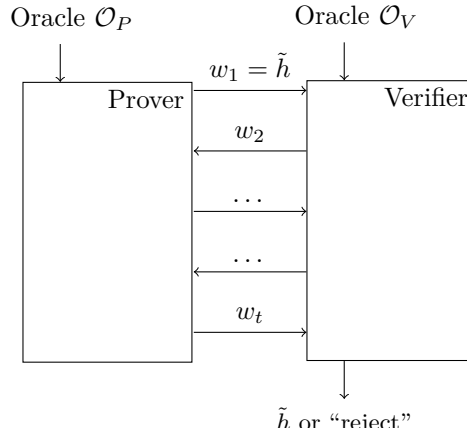
It has been claimed that many or most published research findings are false [14]. Others refer to an ongoing *replication crisis* [20, 7], where many scientific studies are hard or impossible to replicate or reproduce [21, 3]. Addressing these issues is a scientific and societal priority.

There are many factors contributing to this problem, including: structural incentives faced by researchers, scientific journals, referees, and funding bodies; the level of statistical expertise among researchers and referees; differences in the sources of data used for studies and their replication attempts; choice of standards of statistical significance; and norms pertaining to the publication of detailed replicable experimental procedures and complete datasets of experimental results.

We stress that the current paper does not touch on the majority of these issues, and our discussion of the replication crisis (as well as our choice of quotation at the beginning of the paper) does *not* by any means suggest that adoption of PAC verification protocols will single-handedly solve all issues pertaining to replication. Rather, the contribution of the current paper with respect to scientific replication is very specific: we suggest that for some specific types of experiments, PAC verification can be used to design protocols that allow to verify the results of an experiment in a manner that uses a quantitatively smaller (or otherwise cheaper) set of independent experimental data than would be necessary for a traditional replication that fully repeats the original experiment. In [13, Appendix A] we list four such types of experiments. We argue that devising PAC verification protocols that make scientific replication procedures even modestly cheaper for specific types of experiments is a worthwhile endeavor that could help increase the amount of scientific replication or verification that occurs, and decrease the prevalence of errors that remain undiscovered in the scientific literature.

### 1.3 Our Setting

In this paper we consider the following form of interaction between a verifier and a prover.



■ **Figure 1** The verifier and prover each have access to an oracle, and they exchange messages with each other. Eventually, the verifier outputs a hypothesis, or rejects the interaction. One natural case is where the prover suggests a hypothesis  $\tilde{h}$ , and the verifier either accepts or rejects this suggestion.

Let  $\mathcal{H} \subseteq \{0, 1\}^{\mathcal{X}}$  be a class of hypotheses, and let  $\mathcal{D}$  be a distribution over  $\mathcal{X} \times \{0, 1\}$ . The verifier and the prover each have access to an oracle, denoted  $\mathcal{O}_V$  and  $\mathcal{O}_P$  respectively. In the simplest case, both oracles provide i.i.d. samples from  $\mathcal{D}$ . That is, each time an oracle is accessed, it returns a sample from  $\mathcal{D}$  taken independently of all previous samples and

events. In addition, the verifier and prover each have access to a (private) random coin value, denoted  $\rho_V$  and  $\rho_P$  respectively, which are sampled from some known distributions over  $\{0, 1\}^*$  independently of each other and of all other events. During the interaction, the prover and verifier take turns sending each other messages  $w_1, w_2, \dots$ , where  $w_i \in \{0, 1\}^*$  for all  $i$ . Finally, at some point during the exchange of messages,  $V$  halts and outputs either “reject” or a hypothesis  $h : \mathcal{X} \rightarrow \{0, 1\}$ . The goal of the verifier is to output an  $\varepsilon$ -good hypothesis, meaning that

$$L_{\mathcal{D}}(h) \leq L_{\mathcal{D}}(\mathcal{H}) + \varepsilon.$$

A natural special case of interest is when the prover’s and verifier’s oracles provide sample access to  $\mathcal{D}$ . The prover can learn a “good” hypothesis  $\tilde{h} : \mathcal{X} \rightarrow \{0, 1\}$  and send it to the verifier as its first message, as in Figure 1 above. The prover and verifier then exchange further messages, wherein the prover tries to convince the verifier that  $\tilde{h}$  is  $\varepsilon$ -good, and the verifier tries to assess the veracity of that claim. If the verifier is convinced, it outputs  $\tilde{h}$ , otherwise it rejects.

We proceed with an informal definition of PAC verification (see full definitions in Section 1.5). Before doing so, we first recall a relaxed variant of PAC learning, called *semi-agnostic* PAC learning, where we allow a multiplicative slack of  $\alpha \geq 1$  in the error guarantee.

► **Definition** ( $\alpha$ -PAC Learnability – informal version of [13, Definition 1.22]). *A class of hypothesis  $\mathcal{H}$  is  $\alpha$ -PAC learnable (or semi-agnostic PAC learnable with parameter  $\alpha$ ) if there exists an algorithm  $A$  such that for every distribution  $\mathcal{D}$  and every  $\varepsilon, \delta > 0$ , with probability at least  $1 - \delta$ ,  $A$  outputs  $h$  that satisfies*

$$L_{\mathcal{D}}(h) \leq \alpha \cdot L_{\mathcal{D}}(\mathcal{H}) + \varepsilon. \quad (2)$$

PAC verification is the corresponding notion for interactive proof systems:

► **Definition** ( $\alpha$ -PAC Verifiability – informal version of Definition 4). *A class of hypothesis  $\mathcal{H}$  is  $\alpha$ -PAC verifiable if there exists a pair of algorithms  $(P, V)$  that satisfy the following conditions for every distribution  $\mathcal{D}$  and every  $\varepsilon, \delta > 0$ :*

- **Completeness.** *After interacting with  $P$ ,  $V$  outputs  $h$  such that with probability at least  $1 - \delta$ ,  $h \neq \text{reject}$  and  $h$  satisfies (2).*
- **Soundness.** *After interacting with any (possibly unbounded) prover  $P'$ ,  $V$  outputs  $h$  such that with probability at least  $1 - \delta$ , either  $h = \text{reject}$  or  $h$  satisfies (2).*

► **Remark 1.** We insist on double efficiency; that is, that the sample complexity and running times of both  $V$  and  $P$  must be polynomial in  $\frac{1}{\varepsilon}$ ,  $\log(\frac{1}{\delta})$ , and perhaps also in some parameters that depend on  $\mathcal{H}$ , such as the VC dimension or Fourier sparsity of  $\mathcal{H}$ . ┐

## 1.4 Overview of Results

In this paper, we start charting the landscape of machine learning problems with respect to Questions 1 and 2 mentioned above. First, in Section 2 we provide evidence for an affirmative answer to Questions 2. We show an interactive proof system that efficiently verifies the class of Fourier-sparse boolean functions, where the prover uses an oracle that provides query access, and the verifier uses an oracle that only provides random samples. In this proof system, both the verifier and prover send and receive messages.

The class of Fourier-sparse functions is very broad, and includes decision trees, bounded-depth boolean circuits and many other important classes of functions. Moreover, the result is interesting because it supplements the widely-held learning parity with noise (LPN) assumption, which entails that PAC learning this class from random samples alone without the help of a prover is hard [5, 24].

► **Lemma** (Informal version of Lemma 14). *Let  $\mathcal{H}$  be the class of boolean functions  $\{0, 1\}^n \rightarrow \mathbb{R}$  that are  $t$ -sparse.<sup>6</sup> Then  $\mathcal{H}$  is 1-PAC verifiable with respect to the uniform distribution using a verifier that has access only to random samples of the form  $(x, f(x))$ , and a prover that has query access to  $f$ . The verifier in this protocol is not proper; the output is not necessarily  $t$ -sparse, but it is  $\text{poly}(n, t)$ -sparse. The number of samples used by the verifier, the number of queries made by the prover, and their running times are all bounded by  $\text{poly}(n, t, \log(\frac{1}{\delta}), \frac{1}{\epsilon})$ .*

**Proof Idea.** The proof uses two standard tools, albeit in a less-standard way. The first standard tool is the Kushilevitz-Mansour algorithm [16], which can PAC learn any  $t$ -sparse function using random samples, but only if the set of non-zero Fourier coefficients is *known*. The second standard tool is the Goldreich-Levin algorithm [10] and [8, Section 2.5.2.3], which can identify the set of non-zero Fourier coefficients, but requires *query access* in order to do so. The protocol combines the two tools in a manner that overcomes the limitations of each of them. First, the verifier executes the Goldreich-Levin algorithm, but whenever it needs to query the target function, it requests that the prover perform the query and send back the result. However, the verifier cannot trust the prover, and so the verifier engineers the queries in such a way that the answers to a certain random subset of the queries are known to the verifier based on its random sample access. This allows the verifier to detect dishonest provers. When the Goldreich-Levin algorithm terminates and outputs the set of non-zero coefficients, the verifier then feeds them as input to the Kushilevitz-Mansour algorithm to find an  $\epsilon$ -good hypothesis using its random sample access. ◀

In [13, Section 3] we formally answer Question 1 affirmatively by showing that a certain simple class of functions (generalized thresholds) exhibits a quadratic gap in sample complexity between learning and verifying:

► **Lemma** (Informal version of Lemma 15). *There exists a sequence of classes of functions  $\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \dots \subseteq \{0, 1\}^{\mathbb{R}}$  such that for any fixed  $\epsilon, \delta \in (0, \frac{1}{2})$ :*

- (i) *The class  $\mathcal{T}_d$  is proper 2-PAC verifiable, where both the verifier and prover have access to random samples, and the verifier requires only  $\tilde{O}(\sqrt{d})$  samples. Moreover, both the prover and verifier are efficient.*
- (ii) *PAC learning the class  $\mathcal{T}_d$  requires  $\Omega(d)$  samples.*

At this point, a perceptive reader would be justified in raising the following challenges. Perhaps 2-PAC verification requires less samples than 1-PAC learning simply because of the multiplicative slack factor of 2? Alternatively, perhaps the separation follows trivially from property testing results: maybe it is possible to achieve 2-PAC verification simply by having the verifier perform closeness testing using random samples, without needing the help of the prover except for finding the candidate hypothesis? The second part of the lemma dismisses both of these concerns.

---

<sup>6</sup> A function  $f : \{0, 1\}^n \rightarrow \mathbb{R}$  is  $t$ -sparse if it has at most  $t$  non-zero Fourier coefficients, namely  $|\{S \subseteq [n] : \hat{f}(S) \neq 0\}| \leq t$ . See preliminaries in [13, Section 1.6.3] for further details.

- **Lemma** (Informal version of Lemma 15 – Continued). *Furthermore, for any fixed  $\varepsilon, \delta \in (0, \frac{1}{2})$ :*
- (iii) *2-PAC learning the class  $\mathcal{T}_d$  requires  $\tilde{\Omega}(d)$  samples. This is true even if we assume that  $L_{\mathcal{D}}(\mathcal{T}_d) > 0$ , where  $\mathcal{D}$  is the underlying distribution.<sup>7</sup>*
  - (iv) *Testing whether  $L_{\mathcal{D}}(\mathcal{T}_d) \leq \alpha$  or  $L_{\mathcal{D}}(\mathcal{T}_d) \geq \beta$  for any  $0 < \alpha < \beta < \frac{1}{2}$  with success probability at least  $1 - \delta$  when  $\mathcal{D}$  is an unknown distribution (without the help of a prover) requires  $\tilde{\Omega}(d)$  random samples from  $\mathcal{D}$ .*

**Proof Idea.** (ii) follows from a standard application of [13, Theorem 1.13], because  $\text{VC}(\mathcal{T}_d) = d$ . (iii) follows by a reduction from (iv). We prove (iv) by showing a further reduction from the problem of approximating the support size of a distribution, and applying a lower bound for that problem (see [13, Theorem 3.20]).

For (i), recall from the introduction that the difficulty in designing a PAC verification proof system revolves around convincing the verifier that the term  $L_{\mathcal{D}}(\mathcal{H})$  in Equation (1) is large. Therefore, we design our class  $\mathcal{T}_d$  such that it admits a simple *certificate of loss*, which is a string that helps the verifier ascertain that  $L_{\mathcal{D}}(\mathcal{H}) \geq \ell$  for some value  $\ell$ .

To see how that works, first consider the simple class  $\mathcal{T}_1$  of monotone increasing threshold functions  $\mathbb{R} \rightarrow \{0, 1\}$ , as in Figure 5a on page 17 below. Observe that if there are two events  $A = [0, a) \times \{1\}$  and  $B = [b, 1] \times \{0\}$  such that  $a \leq b$  and  $\mathcal{D}(A) = \mathcal{D}(B) = \ell$ , then it must be the case that  $L_{\mathcal{D}}(\mathcal{T}_1) \geq \ell$ . This is true because  $a \leq b$ , and so if a monotone increasing threshold classifies any point in  $A$  correctly it must classify all point in  $B$  incorrectly. Furthermore, if the prover sends a description of  $A$  and  $B$  to the verifier, then the verifier can check, using a constant number of samples, that each of these events has weight approximately  $\ell$  with high probability.

This type of certificate of loss can be generalized to the class  $\mathcal{T}_d$ , in which each function is a concatenation of  $d$  monotone increasing thresholds. A certificate of loss for  $\mathcal{T}_d$  is simply a set of  $d$  certificates of loss  $\{A_i, B_i\}_{i=1}^d$ , one for each of the  $d$  thresholds. The question that arises at this point is how can the verifier verify  $d$  separate certificates while using only  $\tilde{O}(\sqrt{d})$  samples. This is performed using tools from distribution testing: the verifier checks whether the distribution of “errors” in the sets specified by the certificates is close to the prover’s claims. I.e., whether the “weight” of 1-labels in each  $A_i$  and 0-labels in each  $B_i$  in the actual distribution, are close to the weights claimed by the prover. Using an identity tester for distributions this can be done using  $O(\sqrt{d})$  samples (note that the identity tester need not be tolerant!). See [13, Theorem E.1] for further details. ◀

In contrast, in [13, Section 4] we show that verification is not always easier than learning:

► **Lemma** (Informal version of [13, Lemma 4.1]). *There exists a sequence of classes  $\mathcal{H}_1, \mathcal{H}_2, \dots$  such that:*

- *It is possible to PAC learn the class  $\mathcal{H}_d$  using  $\tilde{O}(d)$  samples.*
- *For any interactive proof system that properly 1-PAC verifies  $\mathcal{H}_d$ , in which the verifier uses an oracle providing random samples, the verifier must use at least  $\Omega(d)$  samples.*

► **Remark 2.** The lower bound on the sample complexity of the verifier holds regardless of what oracle is used by the prover. ┘

**Proof Idea.** We specify a set  $\mathcal{X}$  of cardinality  $\Omega(d^2)$ , and take  $\mathcal{H}_d$  to be a randomly-chosen subset of all the balanced functions  $\mathcal{X} \rightarrow \{0, 1\}$  (i.e., functions  $f$  such that  $|f^{-1}(0)| = |f^{-1}(1)|$ ). The sample complexity of PAC learning  $\mathcal{H}_d$  follows from its VC dimension being

<sup>7</sup> In the case where  $L_{\mathcal{D}}(\mathcal{T}_d) = 0$ , 2-PAC learning is the same as PAC learning, so the stronger lower bound in (ii) applies.

$\tilde{O}(d)$ . For the lower bound, consider proper PAC verifying  $\mathcal{H}_d$  in the special case where the distribution  $\mathcal{D}$  satisfies  $\mathbb{P}_{(x,y) \in \mathcal{D}}[y = 1] = 1$ , but the marginal of  $\mathcal{D}$  on  $\mathcal{X}$  is unknown to the verifier. Because every hypothesis in the class assigns the incorrect label 0 to precisely half of the domain, a hypothesis achieves minimal loss if it assigns the 0 labels to a subset of size  $\frac{|\mathcal{X}|}{2}$  that has minimal weight. Hence, the verifier must learn enough about the distribution to identify a specific subset of size  $\frac{|\mathcal{X}|}{2}$  with weight close to minimal. We show that doing so requires  $\Omega\left(\sqrt{|\mathcal{X}|}\right) = \Omega(d)$  samples.  $\blacktriangleleft$

Finally, in [13, Section 5] we show that in the setting of semi-supervised learning, where unlabeled samples are cheap, it is possible to perform PAC verification such that the verifier requires significantly less labeled samples than are required for learning. This verification uses a technique we call *query delegation*, and is efficient in terms of time complexity whenever there exists an efficient ERM algorithm that PAC learns the class using random samples.

## 1.5 Definition of PAC Verification

In Section 1.3 we informally described the setting of this paper. Here, we complete that discussion by providing a formal definition of PAC verification, which is the main object of study in this paper.

► **Notation 3.** We write  $[V^{\mathcal{O}_V}(x_V), P^{\mathcal{O}_P}(x_P)]$  for the random variable denoting the output of the verifier  $V$  after interacting with a prover  $P$ , when  $V$  and  $P$  receive inputs  $x_V$  and  $x_P$  respectively, and have access to oracles  $\mathcal{O}_V$  and  $\mathcal{O}_P$  respectively. The inputs  $x_V$  and  $x_P$  can specify parameters of the interaction, such as the accuracy and confidence parameters  $\varepsilon$  and  $\delta$ . This random variable takes values in  $\{0, 1\}^{\mathcal{X}} \cup \{\text{reject}\}$ , namely, it is either a function  $\mathcal{X} \rightarrow \{0, 1\}$  or it is the value “reject”. The random variable depends on the (possibly randomized) responses of the oracles, and on the random coins of  $V$  and  $P$ .

For a distribution  $\mathcal{D}$ , we write  $V^{\mathcal{D}}$  (or  $P^{\mathcal{D}}$ ) to denote use of an oracle that provides i.i.d. samples from the distributions  $\mathcal{D}$ . Likewise, for a function  $f$ , we write  $V^f$  (or  $P^f$ ) to denote use of an oracle that provides query access to  $f$ . That is, in each access to the oracle,  $V$  (or  $P$ ) sends some  $x \in \mathcal{X}$  to the oracle, and receives the answer  $f(x)$ .

We also write  $[V(S_V, \rho_V), P(S_P, \rho_P)] \in \{0, 1\}^{\mathcal{X}} \cup \{\text{reject}\}$  to denote the deterministic output of the verifier  $V$  after interacting with  $P$  in the case where  $V$  and  $P$  receive fixed random coin values  $\rho_V$  and  $\rho_P$  respectively, and receive fixed samples  $S_V$  and  $S_P$  from their oracles  $\mathcal{O}_V$  and  $\mathcal{O}_P$  respectively.

We are interested in classes  $\mathcal{H}$  for which an  $\varepsilon$ -good hypothesis can always be verified with high probability via this form of interaction between an efficient prover and verifier, as formalized in the following definition. Note that the following definitions include an additional multiplicative slack parameter  $\alpha \geq 1$  in the error guarantee. This parameter does not exist in the standard definition of PAC learning; the standard definition corresponds to the case  $\alpha = 1$ .

► **Definition 4 ( $\alpha$ -PAC Verifiability).** Let  $\mathcal{H} \subseteq \{0, 1\}^{\mathcal{X}}$  be a class of hypotheses, let  $\mathfrak{D} \subseteq \Delta(\mathcal{X} \times \{0, 1\})$  be some family of distributions, and let  $\alpha \geq 1$ . We say that  $\mathcal{H}$  is  $\alpha$ -PAC verifiable with respect to  $\mathfrak{D}$  using oracles  $\mathcal{O}_V$  and  $\mathcal{O}_P$  if there exists a pair of algorithms  $(V, P)$  that satisfy the following conditions for every input  $\varepsilon, \delta > 0$ :

■ **Completeness.** For any distribution  $\mathcal{D} \in \mathfrak{D}$ , the random variable  $h := [V^{\mathcal{O}_V}(\varepsilon, \delta), P^{\mathcal{O}_P}(\varepsilon, \delta)]$  satisfies

$$\mathbb{P}\left[h \neq \text{reject} \wedge \left(L_{\mathcal{D}}(h) \leq \alpha \cdot L_{\mathcal{D}}(\mathcal{H}) + \varepsilon\right)\right] \geq 1 - \delta.$$

- **Soundness.** For any distribution  $\mathcal{D} \in \mathfrak{D}$  and any (possibly unbounded) prover  $P'$ , the random variable  $h := [V^{\mathcal{O}_V}(\varepsilon, \delta), P'^{\mathcal{O}_P}(\varepsilon, \delta)]$  satisfies

$$\mathbb{P} \left[ h \neq \text{reject} \wedge \left( L_{\mathcal{D}}(h) > \alpha \cdot L_{\mathcal{D}}(\mathcal{H}) + \varepsilon \right) \right] \leq \delta.$$

► **Remark 5.** Some comments about this definition:

- The behavior of the oracles  $\mathcal{O}_V$  and  $\mathcal{O}_P$  may depend on the specific underlying distribution  $\mathcal{D} \in \mathfrak{D}$ , which is unknown to the prover and verifier. For example, they may provide samples from  $\mathcal{D}$ .
- We insist on double efficiency; that is, that the sample complexity and running times of both  $V$  and  $P$  must be polynomial in  $\frac{1}{\varepsilon}$ ,  $\log(\frac{1}{\delta})$ , and perhaps also in some parameters that depend on  $\mathcal{H}$ , such as the VC dimension or Fourier sparsity of  $\mathcal{H}$ .
- If for every  $\varepsilon, \delta > 0$ , and for any (possibly unbounded) prover  $P'$ , the value  $h := [V^{\mathcal{O}_V}(\varepsilon, \delta), P'^{\mathcal{O}_P}(\varepsilon, \delta)]$  satisfies  $h \in \mathcal{H} \cup \{\text{reject}\}$  with probability 1 (i.e.,  $V$  never outputs a function that is not in  $\mathcal{H}$ ), then we say that  $\mathcal{H}$  is proper  $\alpha$ -PAC verifiable, and that the proof system proper  $\alpha$ -PAC verifies  $\mathcal{H}$ . ─

► **Remark 6.** An important type of learning (studied e.g. by Angluin [1] and Kushilevitz and Mansour [16]) is *learning with membership queries with respect to the uniform distribution*. In this setting, the family  $\mathfrak{D}$  consists of distributions  $\mathcal{D}$  such that: (1) the marginal distribution of  $\mathcal{D}$  over  $\mathcal{X}$  is uniform; (2)  $\mathcal{D}$  has a target function  $f : \mathcal{X} \rightarrow \{1, -1\}$  satisfying  $\mathbb{P}_{(x,y) \sim \mathcal{D}}[y = f(x)] = 1$ .<sup>8</sup> In Section 2, we will consider protocols for this type of learning that have the form  $[V^{\mathcal{D}}, P^f]$ , such that the verifier has access to an oracle providing random samples from a distribution  $\mathcal{D} \in \mathfrak{D}$ , and the prover has access to an oracle providing query access to  $f$ , the target function of  $\mathcal{D}$ . This type of protocol models a real-world scenario where  $P$  has qualitatively more powerful access to training data than  $V$ . ─

## 2 Efficient Verification for the Class of Fourier-Sparse Functions

In Section 3 below we show that in some cases verification is strictly easier than learning and closeness testing. The verification protocol presented there has a single round, where the prover simply sends a hypothesis and a proof that it is (approximately) optimal. In this section, we describe a multi-round protocol that demonstrates that interaction is helpful for verification.

The interactive protocol we present PAC verifies the class of *Fourier-sparse functions*. This is a broad class of functions, which includes decision trees, DNF formulas with small clauses, and  $\text{AC}^0$  circuits.<sup>9</sup> Every function  $f : \{0, 1\}^n \rightarrow \mathbb{R}$  can be written as a linear combination  $f = \sum_{T \subseteq [n]} \hat{f}(T) \chi_T$ .<sup>10</sup> In Fourier-sparse functions, only a small number of coefficients are non-zero. Note that according to the learning parity with noise (LPN) assumption [5, 24] it is not possible to learn the Fourier-sparse functions efficiently using random samples only.

An important technicality is that throughout this section we focus solely on PAC verification with respect to families of distributions that have a uniform marginal over  $\{0, 1\}^n$ , and have a target function  $f : \{0, 1\}^n \rightarrow \{1, -1\}$  such that  $\mathbb{P}_{(x,y) \sim \mathcal{D}}[y = f(x)] = 1$ . See further discussion in Remark 6 on page 11. In this setting, in order to learn  $f$  it is sufficient to approximate its heavy Fourier coefficients.

<sup>8</sup> Note that  $f$  is not necessarily a member of  $\mathcal{H}$ , so this is still an *agnostic* (rather than *realizable*) case.

<sup>9</sup> See Mansour [18, Section 5.2.2, Theorems 5.15 and 5.16]. ( $\text{AC}^0$  is the set of functions computable by constant-depth boolean circuits with a polynomial number of AND, OR and NOT gates.)

<sup>10</sup> The real numbers  $\hat{f}(T)$  are called *Fourier coefficients*, and the functions  $\chi_T$  are called *characters*.



► **Notation 7.** Let  $f : \{0, 1\}^n \rightarrow \mathbb{R}$ , and let  $\tau \geq 0$ . The set of  $\tau$ -heavy coefficients of  $f$  is  $\hat{f}^{\geq \tau} = \{T \subseteq [n] : |\hat{f}(T)| \geq \tau\}$ .

Furthermore, approximating a single coefficient is easy given random samples from the uniform distribution [13, Claim 2.11]. There are, however, an exponential number of coefficients, so approximating all of them is not feasible. This is where verification comes in. If the set of heavy coefficients is known, and if the function is Fourier-sparse, then one can efficiently learn the function by approximating that particular set of coefficients. The prover can provide the list of heavy coefficients, and then the verifier can learn the function by approximating these coefficients.

The challenge that remains in designing such a verification protocol is to verify that the provided list of heavy coefficients is correct. If the list contains some characters that are not actually heavy, no harm is done.<sup>11</sup> However, if a dishonest prover omits some of the heavy coefficients from the list, how can the verifier detect this omission? The following result provides an answer to this question.

► **Lemma 8 (Interactive Goldreich-Levin).** *There exists an interactive proof system  $(V, P^*)$  as follows. For every  $n \in \mathbb{N}$ ,  $\delta > 0$ , every  $\tau \geq 2^{-\frac{n}{10}}$ , every function  $f : \{0, 1\}^n \rightarrow \{0, 1\}$ , and every prover  $P$ , let*

$$L_P = [V(S, n, \tau, \delta, \rho_V), P^f(n, \tau, \delta, \rho_P)]$$

be a random variable denoting the output of  $V$  after interacting with the prover  $P$ , which has query access to  $f$ , where  $S = ((x_1, f(x_1)), \dots, (x_m, f(x_m)))$  is a random sample with  $x_1, \dots, x_m$  taken independently and uniformly from  $\{0, 1\}^n$ , and  $\rho_V, \rho_P$  are strings of private random coins.  $L_P$  takes values that are either a collection of subsets of  $[n]$ , or “reject”.

The following properties hold:

- **Completeness.**  $\mathbb{P} \left[ L_{P^*} \neq \text{reject} \wedge \hat{f}^{\geq \tau} \subseteq L_{P^*} \right] \geq 1 - \delta$ .
- **Soundness.** For any (possibly unbounded) prover  $P$ ,

$$\mathbb{P} \left[ L_P \neq \text{reject} \wedge \hat{f}^{\geq \tau} \not\subseteq L_P \right] \leq \delta.$$

- **Double efficiency.** The verifier  $V$  uses at most  $O\left(\frac{n}{\tau} \log\left(\frac{n}{\tau}\right) \log\left(\frac{1}{\delta}\right)\right)$  random samples from  $f$  and runs in time  $\text{poly}\left(n, \frac{1}{\tau}, \log\left(\frac{1}{\delta}\right)\right)$ . The runtime of the prover  $P^*$ , and the number of queries it makes to  $f$ , are at most  $O\left(\frac{n^3}{\tau^5} \log\left(\frac{1}{\delta}\right)\right)$ . Whenever  $L_P \neq \text{reject}$ , the cardinality of  $L_P$  is at most  $O\left(\frac{n^2}{\tau^5} \log\left(\frac{1}{\delta}\right)\right)$ .

All proofs for this section appear in Section 2 in the full version of the paper [13].

► **Remark 9.** In Section 1.5 we defined interactive proof systems specifically for PAC verification. The proof system in Lemma 8 is technically different. The verifier outputs a collection of sets instead of a function, and it satisfies different completeness and soundness conditions. ┘

The verifier  $V$  operates by simulating the Goldreich-Levin (GL) algorithm for finding  $\hat{f}^{\geq \tau}$ . However, the GL algorithm requires query access to  $f$ , while  $V$  has access only to random samples. To overcome this limitation,  $V$  delegates the task of querying  $f$  to the

---

<sup>11</sup> The verifier can approximate each coefficient in the list and discard of those that are not heavy. Alternatively, the verifier can include the additional coefficients in its approximation of the target function, because the approximation improves as the number of estimated coefficients grows (so long as the list is polynomial in  $n$ ).



prover  $P$ , who does have the necessary query access. Because  $P$  is not trusted,  $V$  engineers the set of queries it delegates to  $P$  in such a way that some random subset of them already appear in the sample  $S$  which  $V$  has received as input. This allows  $V$  to independently verify a random subset of the results sent by  $P$ , ensuring that a sufficiently dishonest prover is discovered with high probability.

**The Interactive Goldreich-Levin Protocol.** The verifier for Lemma 8 uses Protocol 2 (IGL), which repeatedly applies Protocol 3 (IGL-ITERATION).

■ **Protocol 2** Interactive Goldreich-Levin:  $\text{IGL}(n, \tau, \delta)$ .

```

V performs the following:
   $r \leftarrow \lceil (\frac{4n}{\tau} + 1) \log(\frac{1}{\delta}) \rceil$ 
  for  $i \in [r]$  do
     $L_i \leftarrow \text{IGL-ITERATION}(n, \tau)$ 
    if  $L_i = \text{reject}$  then
      output reject
   $L \leftarrow \bigcup_{i \in [r]} L_i$ 
  output  $L$ 

```

Lemma 8 follows from two claims: Claim 10 says that if the prover is mostly honest, then the output is correct. Claim 11 says that if the prover is too dishonest, it will be rejected.

▷ **Claim 10 (Completeness of IGL).** Consider an execution of  $\text{IGL-ITERATION}(n, \tau)$  for  $\tau \geq 2^{-\frac{n}{10}}$ . For any prover  $P$  and any randomness  $\rho_P$ , if  $V$  did not reject, and the evaluations provided by  $P$  were mostly honest, in the sense that  $\forall i \in [n] : \mathbb{P}_{x \in H} [\tilde{f}(x \oplus e_i) \neq f(x \oplus e_i)] \leq \frac{\tau}{4}$ , then  $\mathbb{P}[\hat{f}^{\geq \tau} \subseteq L] \geq \frac{1}{2}$ , where the probability is over the sample  $S$  and the randomness  $\rho_V$ . ◀

**Proof Idea.** We show that for every heavy coefficient  $T \in \hat{f}^{\geq \tau}$ ,  $\mathbb{P}[T \notin L] \leq \frac{\tau^2}{4}$ . From a union bound, this is sufficient to prove the claim, because Parseval's identity implies that  $|\hat{f}^{\geq \tau}| \leq \frac{1}{\tau^2}$ . The main observation is that the Goldreich-Levin algorithm is resilient to some *adversarial* noise. To see this, note that if  $T \in \hat{f}^{\geq \tau}$ , then  $\mathbb{P}_{x \in \{0,1\}^n} [f(x) = \ell(x)] \geq \frac{1}{2} + \frac{\tau}{2}$  for the linear function  $\ell(x) = \oplus_{i \in T} x_i$  (or its negation  $\neg \ell$ ). Therefore,  $f$  agrees with  $\ell$  with probability at least  $\frac{1}{2} + \frac{\tau}{4}$ , even if a  $\frac{\tau}{4}$ -fraction of  $f$ 's values are adversarially corrupted. Hence, in the iteration of the outer loop in Step 4 in which  $y_j = \ell(b_j)$  for all  $j \in [k]$ , the majority function will output the correct value of  $f(x^K \oplus e_i) \oplus y^K = \ell(x^K \oplus e_i) \oplus \ell(x^K) = \ell(e_i)$ , and this results in  $T$  being added to the output set  $L$ . Some finer details are discussed in the full version of the paper. ◀

▷ **Claim 11 (Soundness of IGL).** Consider an execution of  $\text{IGL-ITERATION}(n, \tau)$ . For any prover  $P$  and any randomness value  $\rho_P$ , if there exists  $i \in [n]$  for which  $P$  was too dishonest in the sense that  $\mathbb{P}_{x \in H} [\tilde{f}(x \oplus e_i) \neq f(x \oplus e_i)] > \frac{\tau}{4}$ , then  $\mathbb{P}[L = \text{reject}] \geq \frac{\tau}{4n}$ , where the probability is over the sample  $S$  and the randomness  $\rho_V$ . ◀

<sup>12</sup> For any  $j$ ,  $e_j$  is a vector in which the  $j$ -th entry is 1 and all other entries are 0.

■ **Protocol 3** Interactive Goldreich-Levin Iteration: IGL-ITERATION( $n, \tau$ ).

**Assumption:**  $V$  receives a sample  $S = ((x_1, f(x_1)), \dots, (x_m, f(x_m)))$  such that  $m = \lceil \log(\frac{40n}{\tau^4} + 1) \rceil$ , for all  $i \in [m]$ ,  $x_i \in \{0, 1\}^n$  is chosen independently and uniformly, and  $f(x_i) \in \{0, 1\}$ .

1.  $V$  selects  $i^* \in [n]$  uniformly at random, and then sends  $B$  to  $P$ , where  $B = \{b_1, \dots, b_k\} \subseteq \{0, 1\}^n$  is a basis chosen uniformly at random from the set of bases of the subspace  $H = \text{span}(\{x_1 \oplus e_{i^*}, \dots, x_m \oplus e_{i^*}\})$ .<sup>12</sup>
2.  $P$  sends  $V$  the set  $\{(x \oplus e_i, \tilde{f}(x \oplus e_i)) : i \in [n] \wedge x \in H\}$ , where for any  $z$ ,  $\tilde{f}(z)$  is purportedly the value of  $f(z)$  obtained using  $P$ 's query access to  $f$ .
3.  $V$  checks that for all  $i \in [m]$ , the evaluation  $f(x_i)$  provided by  $P$  equals that which appeared in the sample  $S$ . If there are any discrepancies,  $V$  rejects the interaction and terminates. Otherwise:
4. Let  $\mathcal{K} = \{K : \emptyset \subsetneq K \subseteq [k]\}$ .  $V$  Performs the following computation and outputs  $L$ :

```

 $L \leftarrow \emptyset$ 
for  $(y_1, \dots, y_k) \in \{0, 1\}^k$  do
  for  $K \in \mathcal{K}$  do
     $x^K \leftarrow \bigoplus_{i \in K} b_i$ 
     $y^K \leftarrow \bigoplus_{i \in K} y_i$ 
  for  $i \in [n]$  do
     $a_i \leftarrow \text{majority}_{K \in \mathcal{K}} (\tilde{f}(x^K \oplus e_i) \oplus y^K)$ 
  add  $\{i : a_i = 1\}$  and  $\{i : a_i = 0\}$  to  $L$ 
output  $L$ 

```

**Proof Idea.** In Protocol 3, the verifier “hides” the sample it knows in the random subspace  $H \oplus e_{i^*}$  for an index  $i^* \in [n]$  chosen uniformly at random. With probability at least  $\frac{1}{n}$ ,  $i^*$  is an index for which the prover was too dishonest. If that is the case, then in expectation a  $\frac{\tau}{4}$ -fraction of the prover’s answers on the known sample were dishonest, because the known sample is a random subset of  $H \oplus e_{i^*}$ . The claim now follows from Markov’s inequality. ◀

► **Remark 12.** It is possible to run all repetitions of the IGL protocol in parallel such that only 2 messages are exchanged. ┘

**Efficient Verification of Fourier-Sparse Functions.** As a corollary of Lemma 8, we obtain the following lemma, which is an interactive version of the Kushilevitz-Mansour algorithm [16, 17, 18]. It says that the class of  $t$ -sparse boolean functions is efficiently PAC verifiable with respect to the uniform distribution using an interactive proof system (Protocol 4) of the form  $[V^{\mathcal{D}}, P^f]$ , where the prover has query access and the verifier has random samples.

Protocol 4 uses a procedure ESTIMATECOEFFICIENT [13, Algorithm 4] that estimates a single fourier coefficient  $\hat{f}(T)$  up to precision  $\lambda$  with confidence  $\delta$  using  $\left\lceil \frac{2 \ln(2/\delta)}{\lambda^2} \right\rceil$  random samples. Note that the output of Protocol 4 is a function  $h : \{0, 1\}^n \rightarrow \mathbb{R}$ , not necessarily a boolean function.

► **Notation 13.** Let  $\mathcal{X}$  be a finite set. We write  $\mathfrak{D}_{\mathcal{U}}^{\text{func}}(\mathcal{X})$  to denote the set of all distributions  $\mathcal{D}$  over  $\mathcal{X} \times \{1, -1\}$  that have the following two properties:

- The marginal distribution of  $\mathcal{D}$  over  $\mathcal{X}$  is uniform. Namely,  $\sum_{y \in \{1, -1\}} \mathcal{D}((x, y)) = \frac{1}{|\mathcal{X}|}$  for all  $x \in \mathcal{X}$ .
- $\mathcal{D}$  has a target function  $f: \mathcal{X} \rightarrow \{1, -1\}$  satisfying  $\mathbb{P}_{(x, y) \sim \mathcal{D}}[y = f(x)] = 1$ .

► **Lemma 14.** Let  $\mathcal{X} = \{0, 1\}^n$ , and let  $\mathcal{H}$  be the class of functions  $\mathcal{X} \rightarrow \mathbb{R}$  that are  $t$ -sparse.<sup>6</sup> The class  $\mathcal{H}$  is 1-PAC verifiable for any  $\varepsilon \geq 4t \cdot 2^{-\frac{n}{10}}$  with respect to  $\mathfrak{D}_{\mathcal{U}}^{\text{func}}(\mathcal{X})$  by a proof system in which the verifier has access to random samples from a distribution  $\mathcal{D} \in \mathfrak{D}_{\mathcal{U}}^{\text{func}}(\mathcal{X})$ , and the honest prover has oracle access to the target function  $f: \mathcal{X} \rightarrow \{1, -1\}$  of  $\mathcal{D}$ . The running time of both parties is at most poly  $(n, t, \frac{1}{\varepsilon}, \log(\frac{1}{\delta}))$ . The verifier in this protocol is not proper; the output is not necessarily  $t$ -sparse, but it is poly  $(n, t, \frac{1}{\varepsilon}, \log(\frac{1}{\delta}))$ -sparse.

■ **Protocol 4** PAC Verification of  $t$ -Sparse Functions:  $\text{VERIFYFOURIERSPARSE}(n, t, \varepsilon, \delta)$ .

$V$  performs the following:

```

 $\tau \leftarrow \frac{\varepsilon}{4t}$ 
 $L \leftarrow \text{IGL}(n, \tau, \frac{\delta}{2})$ 
if  $L = \text{reject}$  then
  output reject
else
   $\lambda \leftarrow \sqrt{\frac{\varepsilon}{8|L|}}$ 
  for  $T \in L$  do
     $\alpha_T \leftarrow \text{ESTIMATECOEFFICIENT}(T, \lambda, \frac{\delta}{2|L|})$ 
   $h \leftarrow \sum_{T \in L} \alpha_T \chi_T$ 
  output  $h$ 

```

Complete proofs appear in Section 2 in the full version of the paper [13].

### 3 Separation Between Learning, Testing, and PAC Verification

In this section we present the following gap in sample complexity between *learning* and *verification*. Conceptually, the result tells us that at least in some scenarios, delegating a learning task to an untrusted party is worthwhile, because verifying that their final result is correct is significantly cheaper than finding that result ourselves.

► **Lemma 15** (Separation Between Learning, Testing, and Verification). *There exists a sequence of classes of functions  $\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \dots \subseteq \{0, 1\}^{\mathbb{R}}$  such that for any fixed  $\varepsilon, \delta \in (0, \frac{1}{2})$  all of the following hold:*

- (i)  $\mathcal{T}_d$  is proper 2-PAC verifiable, where the verifier uses<sup>13</sup>

$$m_V = O\left(\frac{\sqrt{d} \log(d) \log(\frac{1}{\delta})}{\varepsilon^6}\right)$$

random samples, the honest prover uses

<sup>13</sup>We believe that the dependence of  $m_V$  on  $\varepsilon$  can be improved, see [13, Remark 3.15].

$$m_P = O\left(\frac{d^3 \log^2(d)}{\varepsilon^4} \log\left(\frac{d}{\varepsilon}\right) + \frac{d\sqrt{d} \log(d)}{\varepsilon^2} \log\left(\frac{1}{\delta}\right)\right)$$

random samples, and each of them runs in time polynomial in its number of samples.<sup>14</sup>

- (ii) Agnostic PAC learning  $\mathcal{T}_d$  requires  $\Omega\left(\frac{d+\log(\frac{1}{\delta})}{\varepsilon^2}\right)$  samples.
- (iii) If  $\varepsilon \leq \frac{1}{32}$  then 2-PAC learning the class  $\mathcal{T}_d$  requires  $\Omega\left(\frac{d}{\log(d)}\right)$  samples. This is true even if we assume that  $L_{\mathcal{D}}(\mathcal{T}_d) > 0$ , where  $\mathcal{D}$  is the underlying distribution.
- (iv) Testing whether  $L_{\mathcal{D}}(\mathcal{T}_d) \leq \alpha$  or  $L_{\mathcal{D}}(\mathcal{T}_d) \geq \beta$  for any  $0 < \alpha < \beta < \frac{1}{2}$  with success probability at least  $1 - \delta$  when  $\mathcal{D}$  is an unknown distribution (without the help of a prover) requires  $\Omega\left(\frac{d}{\log(d)}\right)$  random samples from  $\mathcal{D}$ .

Our exposition in this section is partial, and aims only to convey the main ideas of part (i). Complete formal proofs appear in [13, Section 3]. We show an MA-like proof system wherein the prover sends a single message  $(\tilde{h}, \tilde{C}, \tilde{\ell})$  such that allegedly  $\tilde{h}$  is an  $\varepsilon$ -good hypothesis with loss at most  $\tilde{\ell} > 0$ .  $\tilde{C} \in \{0, 1\}^*$  is a string called a *certificate of loss*, which helps the verifier ascertain that  $\mathcal{H}$  has a large loss with respect to the unknown distribution  $\mathcal{D}$ . The verifier operates as follows:<sup>15</sup>

- Verifies that  $L_{\mathcal{D}}(\tilde{h}) \leq \tilde{\ell}$  with high probability. That is, it estimates the loss of  $\tilde{h}$  with respect to  $\mathcal{D}$ , and checks that with high probability it is at most  $\tilde{\ell}$ .
- Uses the certificate of loss  $\tilde{C}$  to verify that with high probability,  $L_{\mathcal{D}}(\mathcal{H}) \geq \tilde{\ell} - \varepsilon$ . This step is called *verifying the certificate*.

The class  $\mathcal{T}_d$  of *multi-thresholds* that satisfies Lemma 15 is illustrated in Figure 5, and is defined as follows.

► **Definition 16.** For any  $d \in \mathbb{N}$ , denote by  $\mathcal{T}_d$  the class of functions  $\mathcal{T}_d = \{f_{t_1, \dots, t_d} : t_1, \dots, t_d \in \mathbb{R}\}$  where for all  $t_1, \dots, t_d \in \mathbb{R}$  and  $x \in [0, d]$ , the function  $f_{t_1, \dots, t_d} : \mathbb{R} \rightarrow \{0, 1\}$  is given by

$$f_{t_1, \dots, t_d}(x) = \begin{cases} 0 & x < t_{\lceil x \rceil} \\ 1 & x \geq t_{\lceil x \rceil}, \end{cases}$$

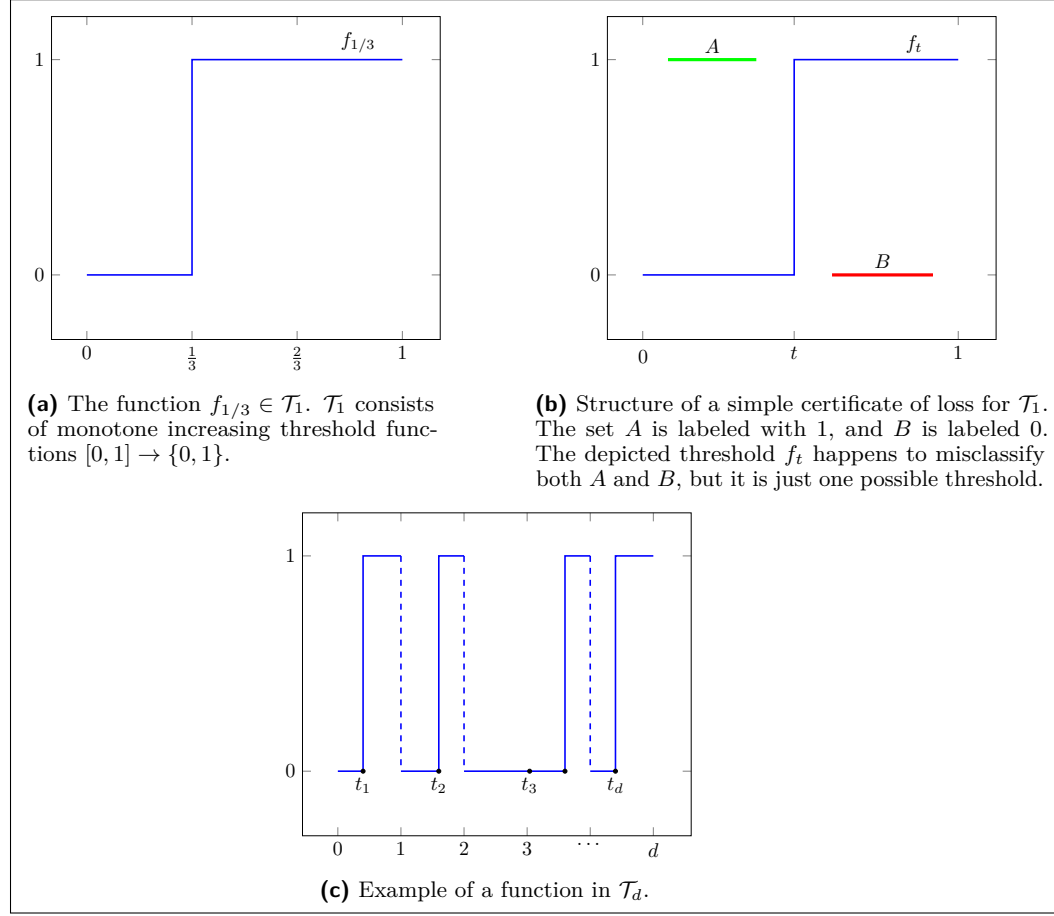
and  $f_{t_1, \dots, t_d}$  vanishes on the complement of  $[0, d]$ .

► **Remark 17.** For convenience, we present the separation result with respect to functions defined over  $\mathbb{R}$ , we assume that the marginal distribution of the samples on  $\mathbb{R}$  is absolutely continuous with respect to the Lebesgue measure, and we ignore issues relating to the representation of real numbers in computations and protocol messages. This provides for a smoother exposition of the ideas. In [13, Appendix C], we show how the results can be discretized. ┘

We first present the certificate structure for the class of threshold functions, namely  $\mathcal{T}_d$  with  $d = 1$ . Certificates of loss for  $\mathcal{T}_1$  are easy to visualize, and they induce a proof system for PAC verifying  $\mathcal{T}_1$  that is complete, sounds, and doubly efficient. However, verifying certificates for  $\mathcal{T}_1$  requires as much resources as PAC learning  $\mathcal{T}_1$  without the help of a prover. The next step is to show that these certificates generalize to the class  $\mathcal{T}_d$  of multi-thresholds, and that for  $\mathcal{T}_d$  there indeed is a gap in sample complexity between verifying and learning.

<sup>14</sup> Subsequent unpublished work by JS and Saachi Mutreja suggests that it is possible to strengthen this to obtain 1-PAC verification with better sample complexity bounds.

<sup>15</sup> We provide a more detailed description of the verification procedure below, and in [13, Claim 3.14].



■ **Figure 5** The class  $\mathcal{T}_d$  of multi-thresholds, with the special case  $\mathcal{T}_1$  and its certificate structure.

**Certificates of loss for  $\mathcal{T}_1$ .** Consider two sets  $A \subseteq [0, 1] \times \{1\}$  and  $B \subseteq [0, 1] \times \{0\}$ , such that all the points in  $A$  are located to the left of all the points in  $B$ , as in Figure 5b. Because we only allow thresholds that are monotone increasing, a threshold that labels any point in  $A$  correctly must label all points of  $B$  incorrectly, and vice versa. Hence, any threshold must have loss at least  $\min\{\mathcal{D}(A), \mathcal{D}(B)\}$ . Estimating  $\mathcal{D}(A)$  and  $\mathcal{D}(B)$  is easy (by Hoeffding's inequality). Formally:

► **Definition 18.** Let  $\mathcal{D} \in \Delta([0, 1] \times \{0, 1\})$  be a distribution and  $\ell, \eta \geq 0$ . A certificate of loss at least  $\ell$  for class  $\mathcal{T}_1$  is a pair  $(a, b)$  where  $0 < a \leq b < 1$ .

We say that the certificate is  $\eta$ -valid with respect to distribution  $\mathcal{D}$  if the events  $A = [0, a] \times \{1\}$  and  $B = [b, 1] \times \{0\}$  satisfy  $|\mathcal{D}(A) - \ell| + |\mathcal{D}(B) - \ell| \leq \eta$ .

▷ **Claim 19** ([13, Claims 3.5 and 3.4]). Let  $\mathcal{D} \in \Delta([0, 1] \times \{0, 1\})$  be a distribution and  $\ell, \eta \geq 0$ .

- Soundness. If  $\mathcal{D}$  has a certificate of loss at least  $\ell$  which is  $\eta$ -valid with respect to  $\mathcal{D}$ , then  $L_{\mathcal{D}}(\mathcal{T}_1) \geq \ell - \eta$ .
- Completeness. If  $L_{\mathcal{D}}(\mathcal{T}_1) = \ell$  then there exists a 0-valid certificate of loss at least  $\frac{\ell}{2}$  with respect to  $\mathcal{D}$ . ┘

**Certificates of loss for  $\mathcal{T}_d$  with  $d > 1$ .** A certificate of loss is simply a collection of  $d$  certificates of loss for  $\mathcal{T}_1$ , one for each unit interval in  $[0, d]$ . Standard techniques from VC theory show that it is possible to efficiently generate certificates for  $\mathcal{T}_d$  for a particular distribution using  $\tilde{O}(d^2)$  samples [13, Claim 3.13]. This is more expensive than learning the class  $\mathcal{T}_d$ , but it may be worthwhile seeing as the verifier can apply techniques from distribution testing to verify purported certificates using only  $\tilde{O}(\sqrt{d})$  samples – which is cheaper than learning [13, Claim 3.14].

Further discussion and complete proofs of Lemma 15(ii)-(iv), including the lower bound for closeness testing, appear in the full version of the paper [13].

## 4 Directions for Future Work

This work initializes the study of verification in the context of machine learning. We have seen separations between the sample complexity of verification versus learning and testing, a protocol that uses interaction to efficiently learn sparse boolean functions, and have seen that in some cases the sample complexities of verification and learning are the same.

Building a theory that can help guide verification procedures is a main objective for future research. A specific approach is to identify dimension-like quantities that describe the sample complexity of verification, similarly to role VC dimension plays in characterizing learnability. A different approach is to understand the trade-offs between the various resources in the system – the amount of time, space and samples used by the prover and the verifier, as well as the amount of interaction between the parties.

From a practical perspective, we described potential applications for delegation of machine learning, and for verification of experimental data. It seems beneficial to build efficient verification protocols for machine learning problems that are commonly used in practice, and for the types of scientific experiments mentioned in [13, Appendix A]. This would have commercial and scientific applications.

There are also some technical improvements that we find interesting. For example, is there a simple way to improve the MA-like protocol for the multi-thresholds class  $\mathcal{T}_d$  to achieve 1-PAC verification (instead of 2-PAC verification)?

Finally, seeing as learning verification is still a new concept, it would be good to consider alternative formal definitions, investigate how robust our definition is, and discuss what the “right” definition should be.

Additional directions are discussed in Section 6 in the full version of the paper [13]. ◀

---

## References

- 1 Dana Angluin. Learning regular sets from queries and counterexamples. *Inf. Comput.*, 75(2):87–106, 1987. doi:10.1016/0890-5401(87)90052-6.
- 2 Maria-Florina Balcan, Eric Blais, Avrim Blum, and Liu Yang. Active property testing. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 21–30. IEEE Computer Society, 2012. doi:10.1109/FOCS.2012.64.
- 3 C. Glenn Begley and Lee M. Ellis. Raise standards for preclinical cancer research. *Nature*, 483(7391):531–533, 2012.
- 4 Avrim Blum and Lunjia Hu. Active tolerant testing. In Sébastien Bubeck, Vianney Perchet, and Philippe Rigollet, editors, *Conference On Learning Theory, COLT 2018, Stockholm, Sweden, 6-9 July 2018*, volume 75 of *Proceedings of Machine Learning Research*, pages 474–497. PMLR, 2018. URL: <http://proceedings.mlr.press/v75/blum18a.html>.

- 5 Avrim Blum, Adam Kalai, and Hal Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. *Journal of the ACM (JACM)*, 50(4):506–519, 2003.
- 6 Alessandro Chiesa and Tom Gur. Proofs of proximity for distribution testing. In *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, pages 53:1–53:14, 2018. doi:10.4230/LIPIcs.ITCS.2018.53.
- 7 Fiona Fidler and John Wilcox. Reproducibility of scientific results. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, winter 2018 edition, 2018.
- 8 Oded Goldreich. *Foundations of cryptography: volume 1, basic tools*. Cambridge university press, 2007.
- 9 Oded Goldreich, Shafi Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *J. ACM*, 45(4):653–750, 1998. doi:10.1145/285055.285060.
- 10 Oded Goldreich and Leonid A Levin. A hard-core predicate for all one-way functions. In *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 25–32. ACM, 1989.
- 11 Shafi Goldwasser, Yael Tauman Kalai, and Guy N. Rothblum. Delegating computation: Interactive proofs for muggles. *J. ACM*, 62(4):27:1–27:64, 2015. doi:10.1145/2699436.
- 12 Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on computing*, 18(1):186–208, 1989.
- 13 Shafi Goldwasser, Guy N. Rothblum, Jonathan Shafer, and Amir Yehudayoff. Interactive proofs for verifying machine learning. *Electron. Colloquium Comput. Complex.*, 27:58, 2020. URL: <https://eccc.weizmann.ac.il/report/2020/058>.
- 14 John PA Ioannidis. Why most published research findings are false. *PLoS medicine*, 2(8):e124, 2005.
- 15 Michael J. Kearns and Dana Ron. Testing problems with sublearning sample complexity. *J. Comput. Syst. Sci.*, 61(3):428–456, 2000. doi:10.1006/jcss.1999.1656.
- 16 Eyal Kushilevitz and Yishay Mansour. Learning decision trees using the Fourier spectrum. *SIAM Journal on Computing*, 22(6):1331–1348, 1993.
- 17 Nathan Linial, Yishay Mansour, and Noam Nisan. Constant depth circuits, Fourier transform, and learnability. *Journal of the ACM (JACM)*, 40(3):607–620, 1993.
- 18 Yishay Mansour. Learning boolean functions via the Fourier transform. In *Theoretical advances in neural computation and learning*, pages 391–424. Springer, 1994.
- 19 Michal Parnas, Dana Ron, and Ronitt Rubinfeld. Tolerant property testing and distance approximation. *J. Comput. Syst. Sci.*, 72(6):1012–1042, 2006. doi:10.1016/j.jcss.2006.03.002.
- 20 Harold Pashler and Eric-Jan Wagenmakers. Editors’ introduction to the special section on replicability in psychological science: A crisis of confidence? *Perspectives on Psychological Science*, 7(6):528–530, 2012.
- 21 Florian Prinz, Thomas Schlange, and Khusru Asadullah. Believe it or not: how much can we rely on published data on potential drug targets? *Nature reviews Drug discovery*, 10(9):712–712, 2011.
- 22 Leslie G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.
- 23 Michael Walfish and Andrew J. Blumberg. Verifying computations without reexecuting them. *Commun. ACM*, 58(2):74–84, 2015. doi:10.1145/2641562.
- 24 Yu Yu and John Steinberger. Pseudorandom functions in almost constant depth from low-noise LPN. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 154–183. Springer, 2016.





# Ordered Graph Limits and Their Applications

**Omri Ben-Eliezer**

Center of Mathematical Sciences and Applications, Harvard University, Cambridge, MA, USA  
omribene@cmsa.fas.harvard.edu

**Eldar Fischer**

Faculty of Computer Science, Technion - Israel Institute of Technology, Haifa, Israel  
eldar@cs.technion.ac.il

**Amit Levi**

Cheriton School of Computer Science, University of Waterloo, Canada  
amit.levi@uwaterloo.ca

**Yuichi Yoshida**

Principles of Informatics Research Division, National Institute of Informatics (NII), Tokyo, Japan  
yyoshida@nii.ac.jp

---

## Abstract

The emerging theory of graph limits exhibits an analytic perspective on graphs, showing that many important concepts and tools in graph theory and its applications can be described more naturally (and sometimes proved more easily) in analytic language. We extend the theory of graph limits to the ordered setting, presenting a limit object for dense vertex-ordered graphs, which we call an *orderon*. As a special case, this yields limit objects for matrices whose rows and columns are ordered, and for dynamic graphs that expand (via vertex insertions) over time. Along the way, we devise an ordered locality-preserving variant of the cut distance between ordered graphs, showing that two graphs are close with respect to this distance if and only if they are similar in terms of their ordered subgraph frequencies. We show that the space of orderons is compact with respect to this distance notion, which is key to a successful analysis of combinatorial objects through their limits. For the proof we combine techniques used in the unordered setting with several new techniques specifically designed to overcome the challenges arising in the ordered setting.

We derive several applications of the ordered limit theory in extremal combinatorics, sampling, and property testing in ordered graphs. In particular, we prove a new ordered analogue of the well-known result by Alon and Stav [RS&A'08] on the furthest graph from a hereditary property; this is the first known result of this type in the ordered setting. Unlike the unordered regime, here the Erdős-Rényi random graph  $G(n, p)$  with an ordering over the vertices is *not* always asymptotically the furthest from the property for some  $p$ . However, using our ordered limit theory, we show that random graphs generated by a stochastic block model, where the blocks are consecutive in the vertex ordering, are (approximately) the furthest. Additionally, we describe an alternative analytic proof of the ordered graph removal lemma [Alon et al., FOCS'17].

**2012 ACM Subject Classification** Mathematics of computing → Functional analysis; Mathematics of computing → Nonparametric representations; Mathematics of computing → Extremal graph theory; Theory of computation → Streaming, sublinear and near linear time algorithms

**Keywords and phrases** graph limits, ordered graph, graphon, cut distance, removal lemma

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.42

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/1811.02023>.

**Funding** *Omri Ben-Eliezer*: Part of this work was done while the author was at Tel Aviv University. *Amit Levi*: Research supported by the David R. Cheriton Graduate Scholarship. Part of this work was done while the author was visiting the Technion.

*Yuichi Yoshida*: Research supported by JSPS KAKENHI Grant Number JP17H04676.



© Omri Ben-Eliezer, Eldar Fischer, Amit Levi, and Yuichi Yoshida;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 42; pp. 42:1–42:20



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

## 1 Introduction

Large graphs appear in many applications across all scientific areas. Naturally, it is interesting to try to understand their structure and behavior: When can we say that two graphs are similar (even if they do not have the same size)? How can the convergence of graph sequences be defined? What properties of a large graph can we capture by taking a small sample from it?

The theory of graph limits addresses such questions from an analytic point of view. The investigation of convergent sequences of dense graphs was started to address three seemingly unrelated questions asked in different fields: statistical physics, theory of networks and the Internet, and quasi-randomness. A comprehensive series of papers [12, 13, 28, 21, 29, 14, 11, 30, 15] laid the infrastructure for a rigorous study of the theory of dense graph limits, demonstrating various applications in many areas of mathematics and computer science. The book of Lovász on graph limits [27] presents these results in a unified form.

A sequence  $\{G_n\}_{n=1}^\infty$  of finite graphs, whose number of vertices tends to infinity as  $n \rightarrow \infty$ , is considered *convergent*<sup>1</sup> if the frequency<sup>2</sup> of any fixed graph  $F$  as a subgraph in  $G_n$  converges as  $n \rightarrow \infty$ . The limit object of a convergent sequence of (unordered) graphs in the dense setting, called a *graphon*, is a measurable symmetric function  $W: [0, 1]^2 \rightarrow [0, 1]$ , and it was proved in [28] that, indeed, for any convergent sequence  $\{G_n\}$  of graphs there exists a graphon serving as the limit of  $G_n$  in terms of subgraph frequencies. Apart from their role in the theory of graph limits, graphons are useful in probability theory, as they give rise to exchangeable random graph models; see e.g. [17, 33]. An analytic theory of convergence has been established for many other types of discrete structures. These include sparse graphs, for which many different (and sometimes incomparable) notions of limits exist – see e.g. [16, 10] for two recent papers citing and discussing many of the works in this field; permutations, first developed in [25] and further investigated in several other works; partial orders [26]; and high dimensional functions over finite fields [35]. The limit theory of dense graphs has also been extended to hypergraphs, see [36, 18] and the references within.

In this work we extend the theory of dense graph limits to the ordered setting, establishing a limit theory for vertex-ordered graphs in the dense setting, and presenting several applications of this theory. An *ordered graph* is a symmetric function  $G: [n]^2 \rightarrow \{0, 1\}$ .  $G$  is *simple* if  $G(x, x) = 0$  for any  $x$ . A *weighted ordered graph* is a symmetric function  $F: [n]^2 \rightarrow [0, 1]$ . Unlike the unordered setting, where  $G, G': [n]^2 \rightarrow \Sigma$  are considered isomorphic if there is a permutation  $\pi$  over  $[n]$  so that  $G(x, y) = G'(\pi(x), \pi(y))$  for any  $x \neq y \in [n]$ , in the ordered setting, the automorphism group of a graph  $G$  is trivial:  $G$  is only isomorphic to itself through the identity function.

For simplicity, we consider in the following only graphs (without edge colors). All results here can be generalized in a relatively straightforward manner to edge-colored graph-like ordered structures, where pairs of vertices may have one of  $r \geq 2$  colors (the definition above corresponds to the case  $r = 2$ ). This is done by replacing the range  $[0, 1]$  with the  $(r - 1)$ -dimensional simplex (corresponding to the set of all possible distributions over  $[r]$ ).

Two interesting special cases of two-dimensional ordered structures for which our results naturally yield a limit object are *images*, i.e., ordered matrices, and *dynamic graphs* with vertex insertions. Specifically, (binary)  $m \times n$  images can be viewed as ordered bipartite

<sup>1</sup> In unordered graphs, this is also called *convergence from the left*; see the discussion on [14].

<sup>2</sup> The frequency of  $F$  in  $G$  is roughly defined as the ratio of induced subgraphs of  $G$  isomorphic to  $F$  among all induced subgraphs of  $G$  on  $|F|$  vertices.

graphs  $I: [m] \times [n] \rightarrow \{0, 1\}$ , and our results can be adapted to get a bipartite ordered limit object for them as long as  $m = \Theta(n)$ . Meanwhile, a dynamic graph with vertex insertions can be viewed as a sequence  $\{G_i\}_{i=1}^\infty$  of ordered graphs, where  $G_{i+1}$  is the result of adding a vertex to  $G_i$  and connecting it to the previous vertices according to some prescribed rule. It is natural to view such dynamic graphs that evolve with time as ordered ones, as the time parameter induces a natural ordering. Thus, our work gives, for example, a limit object for time-series where there are pairwise relations between events occurring at different times.

As we shall see in Subsection 1.2, the main results proved in this paper are, in a sense, natural extensions of results in the unordered setting. However, proving them requires machinery that is heavier than that used in the unordered setting: the tools used in the unordered setting are not rich enough to overcome the subtleties materializing in the ordered setting. In particular, the limit object we use in the ordered setting – which we call an *orderon* – has a 4-dimensional structure that is more complicated than the analogous 2-dimensional structure of the graphon, the limit object for the unordered setting. The tools required to establish the ordered theory are described next.

## 1.1 Main ingredients

Let us start by considering a simple yet elusive sequence of ordered graphs, which has the makings of convergence. The *odd-clique* ordered graph  $H_n$  on  $2n$  vertices is defined by setting  $H_n(i, j) = 1$ , i.e., having an edge between vertices  $i$  and  $j$ , if and only if  $i \neq j$  and  $i, j$  are both odd, and otherwise setting  $H_n(i, j) = 0$ . In this subsection we closely inspect this sequence to demonstrate the challenges arising while trying to establish a theory for ordered graphs, and the solutions we propose for them. First, let us define the notions of subgraph frequency and convergence.

The (induced) frequency  $t(F, G)$  of a simple ordered graph  $F$  on  $k$  vertices in an ordered graph  $G$  with  $n$  vertices is the probability that, if one picks  $k$  vertices of  $G$  uniformly and independently (repetitions are allowed) and reorders them as  $x_1 \leq \dots \leq x_k$ ,  $F$  is isomorphic to the induced ordered subgraph of  $G$  over  $x_1, \dots, x_k$ . (The latter is defined as the ordered graph  $H$  on  $k$  vertices satisfying  $H(i, j) = G(x_i, x_j)$  for any  $i, j \in [k]$ .) A sequence  $\{G_n\}_{n=1}^\infty$  of ordered graphs is *convergent* if  $|V(G_n)| \rightarrow \infty$  as  $n \rightarrow \infty$ , and the frequency  $t(F, G_n)$  of any simple ordered graph  $F$  converges as  $n \rightarrow \infty$ . Observe that the odd-clique sequence  $\{H_n\}$  is indeed convergent: The frequency of the empty  $k$ -vertex graph in  $H_n$  tends to  $(k+1)2^{-k}$  as  $n \rightarrow \infty$ , the frequency of any non-empty  $k$ -vertex ordered graph containing only a clique and a (possibly empty) set of isolated vertices tends to  $2^{-k}$ , and the frequency of any other graph in  $H_n$  is 0.<sup>3</sup>

In light of previous works on the unordered theory of convergence, we look for a limit object for ordered graphs that has the following features.

**Representation of finite ordered graphs.** The limit object should have a natural and consistent representation for finite ordered graphs. As in graphons, we allow graphs  $G$  and  $H$  to have the same representation when one is a blowup<sup>4</sup> of the other.

**Usable distance notion.** Working directly with the definition of convergence in terms of subgraph frequencies is difficult. The limit object we seek should be endowed with a metric, like the cut distance for unordered graphs (see discussion below), that should be easier to work with and must have the following property: A sequence of ordered graph is convergent (in terms of frequencies) if and only if it is Cauchy in the metric.

<sup>3</sup> To see why the sum of frequencies is 1, note that for  $k \geq \ell \geq 2$ , the number of  $k$ -vertex ordered graphs consisting of an  $\ell$ -vertex clique and  $k - \ell$  isolated vertices is  $\binom{k}{\ell}$ .

<sup>4</sup> A graph  $G$  on  $nt$  vertices is an ordered  $t$ -blowup of  $H$  on  $n$  vertices if  $G(x, y) = H(\lceil x/t \rceil, \lceil y/t \rceil)$  for any  $x$  and  $y$ .

**Completeness and compactness.** The space of limit objects must be complete with respect to the metric: Cauchy sequences should converge in this metric space. Combined with the previous requirements, this will ensure that any convergent sequence of ordered graphs has a limit (in terms of ordered frequencies), as desired. It is even better if the space is compact, as compactness is essentially an “ultimately strong” version of Szemerédi’s regularity lemma [34], and will help to develop applications of the theory in other areas.

Additionally, we would like the limit object to be as simple as possible, without unnecessary over-representation. In the unordered setting, the metric used is the *cut distance*, introduced by Frieze and Kannan [22, 23] and defined as follows. First, we define the *cut norm*  $\|W\|_{\square}$  of a function  $W: [0, 1]^2 \rightarrow \mathbb{R}$  as the supremum of  $|\int_{S \times T} W(x, y) dx dy|$  over all measurable subsets  $S, T \subseteq [0, 1]$ . The *cut distance* between graphons  $W$  and  $W'$  is the infimum of  $\|W^{\phi} - W'\|_{\square}$  over all measure-preserving bijections  $\phi: [0, 1] \rightarrow [0, 1]$ , where  $W^{\phi}(x, y) \stackrel{\text{def}}{=} W(\phi(x), \phi(y))$ .

For the ordered setting, we look for a similar metric; the cut distance itself does not suit us, as measure-preserving bijections do not preserve ordered subgraph frequencies in general. A first intuition is then to try graphons as the limit object, endowed with the metric  $d_{\square}(W, W') \stackrel{\text{def}}{=} \|W - W'\|_{\square}$ . However, this metric does not satisfy the second requirement: the odd-clique sequence is convergent, yet it is not Cauchy in  $d_{\square}$ , since  $d_{\square}(H_n, H_{2n}) = 1/2$  for any  $n$ . Seeing that  $d_{\square}$  seems “too strict” as a metric and does not capture the similarities between large odd-clique graphs well, it might make sense to use a slightly more “flexible” metric, which allows for measure-preserving bijections, as long as they do not move any of the points too far from its original location. In view of this, we define the *cut-shift distance* between two graphons  $W, W'$  as

$$d_{\triangle}(W, W') \stackrel{\text{def}}{=} \inf_f (\text{Shift}(f) + \|W^f - W'\|_{\square}), \quad (1)$$

where  $f: [0, 1] \rightarrow [0, 1]$  is a measure-preserving bijection,  $\text{Shift}(f) = \sup_{x \in [0, 1]} |f(x) - x|$ , and  $W^f(x, y) = W(f(x), f(y))$  for any  $x, y \in [0, 1]$ . As we show in this paper (Theorem 2 below), the cut-shift distance settles the second requirement: a sequence of ordered graphs is convergent *if and only if* it is Cauchy in the cut-shift distance.

Consider now graphons as a limit object, coupled with the cut-shift distance as a metric. Do graphons satisfy the third requirement? In particular, does there exist a graphon whose ordered subgraph frequencies are equal to the limit frequencies for the odd-clique sequence? The answers to both of these questions are negative: it can be shown that such a graphon cannot exist in view of Lebesgue’s density theorem, which states that there is no measurable subset of  $[0, 1]$  whose density in every interval  $(a, b)$  is  $(b - a)/2$  (see e.g. Theorem 2.5.1 in the book of Franks on Lebesgue measure [20]). Thus, we need a somewhat richer ordered limit object that will allow us to “bypass” the consequences of Lebesgue’s density theorem. Consider for a moment the graphon representations of the odd clique graphs. In these graphons, the domain  $[0, 1]$  can be partitioned into increasingly narrow intervals that alternately represent odd and even vertices. Intuitively, it seems that our limit object needs to be able to contain infinitesimal odd and even intervals at any given location, leading us to the following limit object candidate, which we call an *orderon*.

An orderon is a symmetric measurable function  $W: ([0, 1]^2)^2 \rightarrow [0, 1]$  viewed, intuitively and loosely speaking, as follows. In each point  $(x, a) \in [0, 1]^2$ , corresponding to an infinitesimal “vertex” of the orderon, the first coordinate,  $x$ , represents a location in the linear order of  $[0, 1]$ . Each set  $\{x\} \times [0, 1]$  can thus be viewed as an infinitesimal probability space of vertices that have the same location in the linear order. The role of the second coordinate is to allow “variability” (in terms of probability) of the infinitesimal “vertex” occupying this point in

the order. The definition of the frequency  $t(F, W)$  of a simple ordered graph  $F = ([k], E)$  in an orderon  $W$  is a natural extension of frequency in graphons. First, define the random variable  $\mathbf{G}(k, W)$  as follows: Pick  $k$  points in  $[0, 1]^2$  uniformly and independently, order them according to the first coordinate as  $(x_1, a_1), \dots, (x_k, a_k)$  with  $x_1 \leq \dots \leq x_k$ , and then return a  $k$ -vertex graph  $G$ , in which the edge between each pair of vertices  $i$  and  $j$  exists with probability  $W((x_i, a_i), (x_j, a_j))$ , independently of other edges. The frequency  $t(F, W)$  is defined as the probability that the graph generated according to  $\mathbf{G}(k, W)$  is isomorphic to  $F$ .

Consider the orderon  $W$  satisfying  $W((x, a), (y, b)) = 1$  if and only if  $a, b \leq 1/2$ , and otherwise  $W((x, a), (y, b)) = 0$ .  $W$  now emerges as a natural limit object for the odd-clique sequence: one can verify that the subgraph frequencies in it are as desired.

The cut-shift distance for orderons is defined similarly to (1), except that  $f$  is now a measure-preserving bijection from  $[0, 1]^2$  to  $[0, 1]^2$  and  $\text{Shift}(f) = \sup_{(x,a) \in [0,1]^2} |\pi_1(f(x, a)) - x|$ , where  $\pi_1(y, b) \stackrel{\text{def}}{=} y$  is the projection to the first coordinate.

## 1.2 Main results

Let  $\mathcal{W}$  denote the space of orderons endowed with the cut-shift distance. In view of Lemma 19 below,  $d_\Delta$  is a pseudo-metric for  $\mathcal{W}$ . By identifying  $W, U \in \mathcal{W}$  whenever  $d_\Delta(W, U) = 0$ , we get a metric space  $\widetilde{\mathcal{W}}$ . The following result is the main component for the viability of our limit object, settling the third requirement above.

► **Theorem 1.** *The space  $\widetilde{\mathcal{W}}$  is compact with respect to  $d_\Delta$ .*

The proof of Theorem 1 is significantly more involved than the proof of its unordered analogue. While at a very high level, the roadmap of the proof is similar to that of the unordered one, our setting induces several new challenges, and to handle them we develop new *shape approximation* techniques. These are presented along the proof of the theorem in Section 4.

The next result shows that convergence in terms of frequencies is equivalent to being Cauchy in  $d_\Delta$ . This settles the second requirement.

► **Theorem 2.** *Let  $\{W_n\}_{n=1}^\infty$  be a sequence of orderons. Then  $\{W_n\}$  is Cauchy in  $d_\Delta$  if and only if  $t(F, W_n)$  converges for any fixed simple ordered graph  $F$ .*

As a corollary of the last two results, we get the following.

► **Corollary 3.** *For every convergent sequence of ordered graphs  $\{G_n\}_{n \in \mathbb{N}}$ , there exists an orderon  $W \in \mathcal{W}$  such that  $t(F, G_n) \rightarrow t(F, W)$  for every ordered graph  $F$ .*

The next main result is a sampling theorem, stating that a large enough sample from an orderon is almost always close to it in cut-shift distance. For this, we define the orderon representation  $W_G$  of an  $n$ -vertex ordered graph  $G$  by setting  $W_G((x, a), (y, b)) = G(Q_n(x), Q_n(y))$  for any  $x, a, y, b$ , where we define  $Q_n(x) = \lceil nx \rceil$  for  $x > 0$  and  $Q_n(0) = 1$ . This addresses the first requirement.

► **Theorem 4.** *Let  $k$  be a positive integer and let  $W \in \mathcal{W}$  be an orderon. Let  $G \sim \mathbf{G}(k, W)$ . Then,*

$$d_\Delta(W, W_G) \leq C \left( \frac{\log \log k}{\log k} \right)^{1/3}$$

*holds with probability at least  $1 - C \exp(-\sqrt{k}/C)$  for some constant  $C > 0$ .*

Theorem 4 implies, in particular, that ordered graphs are a dense subset in  $\mathcal{W}$ .

► **Corollary 5.** *For every orderon  $W$  and every  $\varepsilon > 0$ , there exists a simple ordered graph  $G$  on at most  $2^{\varepsilon^{-3+o(1)}}$  vertices such that  $d_\Delta(W, W_G) \leq \varepsilon$ .*

Our next result asserts that any orderon  $W \in \mathcal{W}$  can be approximated in  $L_1$ -distance by an orderon  $U$  with a finite block structure, with the added property that any ordered finite structure that appears with positive density in  $U$  also has positive density in  $W$ .<sup>5</sup> The orderon  $U$  is described as follows. The point set  $[0, 1]^2$  is divided into  $b$  “blocks”, which are subsets of the form  $[(i-1)/b, i/b] \times [0, 1]$  for some  $i \in [b]$ . Each block is decomposed into  $l$  “layers”, of the form  $[(i-1)/b, i/b] \times [(j-1)/l, j/l]$  where  $j \in [l]$ . The value of  $U((x, a), (y, b))$  is now only dependent on which blocks  $x, y$  belong to, which layers  $a, b$  belong to, and possibly whether  $x < y$ . For example, the orderon  $U$  representing the limit of the odd-clique sequence (defined by  $U((x, a), (y, b)) = 1$  if  $a, b \leq 1/2$ , and  $U((x, a), (y, b)) = 0$  elsewhere) has one block and two layers in it. Roughly speaking, one can think of such  $U$  as the orderon representation of a “pixelized” ordered graph, where each vertex (block) consists of multiple “pixels” (here a pixel corresponds to a block-layer pair), and there is a weighted edge<sup>6</sup> between each pair of pixels. Therefore we call an orderon  $U$  with such structure a *pixelized* orderon and term our result the *pixelization lemma*.

► **Theorem 6** (Pixelization lemma; informal). *For any orderon  $W$  and  $\varepsilon > 0$ , there exists a pixelized orderon  $U$  so that  $d_1(U, W) \leq \varepsilon$ , satisfying the following: for all ordered graphs  $F$  with  $t(F, U) > 0$ , we have  $t(F, W) > 0$ .*

We note that the pixelized structure of  $U$  is necessary for this statement to be correct; it is no longer correct in general if we insist that  $U$  must be the orderon representation of a standard edge-weighted ordered graph.

The pixelization lemma is especially useful for applications where the  $L_1$ -distance comes into play. Two such applications, reproving the ordered graph removal lemma [2] and proving a new result in extremal combinatorics, are described next.

### 1.3 The furthest ordered graph from a hereditary property

Here and in the next subsection we describe three applications of our ordered limit theory. We start with an extensive discussion on the first application: A new result on the maximum edit<sup>7</sup> distance  $d_1(G, \mathcal{H})$  of an ordered graph  $G$  from a hereditary<sup>8</sup> property  $\mathcal{H}$ .

For a hereditary property  $\mathcal{H}$  of simple ordered graphs, define  $\overline{d_{\mathcal{H}}} = \sup_G d_1(G, \mathcal{H})$  where  $G$  ranges over all simple graphs (of any size). The parameter  $d_{\mathcal{H}}$  has been widely investigated for unordered graphs. A well-known surprising result of Alon and Stav [6] states, roughly speaking, that  $d_{\mathcal{H}}$  is always “achieved” by the Erdős-Rényi random graph  $\mathbf{G}(n, p)$  for an appropriate choice of  $p$  and large enough  $n$ .

<sup>5</sup> A weaker result, in which the  $L_1$ -distance is replaced by the cut-shift distance, is not hard to prove using our previous main results; we note that it is indeed strictly weaker since the  $L_1$ -distance between any two orderons  $U$  and  $W$  is always at least as large as (and sometimes much larger than)  $d_\Delta(U, W)$ .

<sup>6</sup> In fact, a weighted bi-directed edge, with possibly different weights in the two different directions.

<sup>7</sup> For our purposes, define the edit (or Hamming) distance between two ordered graphs  $G$  and  $G'$  on  $n$  vertices as the smallest number of entries that one needs to change in the adjacency matrix  $A_G$  of  $G$  to make it equal to  $A_{G'}$ , divided by  $n^2$ . For this matter, the adjacency matrix  $A_G$  of a graph  $G$  over vertices  $v_1 < \dots < v_n$  is a binary  $n \times n$  matrix where  $A_G(i, j) = 1$  if and only if there is an edge between  $v_i$  and  $v_j$  in  $G$ . The distance between  $G$  and a property  $\mathcal{P}$  of ordered graphs is  $\min_{G'} d_1(G, G')$  where  $G'$  ranges over all graphs  $G'$  of the same size as  $G$ . The definition for unordered graphs is similar; the only difference is in the notion of isomorphism.

<sup>8</sup> A property of (ordered or unordered) graphs is *hereditary* if it is closed under taking induced subgraphs.



► **Theorem 7** ([6]). *For any hereditary property  $\mathcal{H}$  of unordered graphs there exists  $p_{\mathcal{H}} \in [0, 1]$  satisfying the following. A graph  $G \sim \mathbf{G}(n, p_{\mathcal{H}})$  satisfies  $d_1(G, \mathcal{H}) \geq \overline{d_{\mathcal{H}}} - o(1)$  with high probability.*

In other words, a random graph  $\mathbf{G}(n, p_{\mathcal{H}})$  is with high probability asymptotically (that is, up to relative edit distance of  $o(1)$ ) the furthest from the property  $\mathcal{H}$ . From the analytic perspective, Lovász and Szegedy [30] were able to reprove (and extend) this result using graph limits.

The surprising result of Alon and Stav has led naturally to a very interesting and highly non-trivial question, now known as the (extremal) *graph edit distance problem* [31], which asks the following: Given a hereditary property of interest  $\mathcal{H}$ , what is the value (or values)  $p_{\mathcal{H}}$  that maximizes the distance of  $\mathbf{G}(n, p)$  from  $\mathcal{H}$ ? The general question of determining  $p_{\mathcal{H}}$  given any  $\mathcal{H}$  is currently wide open, although there have been many interesting developments for various classes of hereditary properties; see [31] for an extensive survey of previous works and useful techniques.

While the situation in unordered graphs, and even in (unordered) directed graphs [7] and matrices [32] has been thoroughly investigated, for ordered graphs no result in the spirit of Theorem 7 is known. The first question that comes to mind is whether the behavior in the ordered setting is similar to that in the unordered case: Is it true that for any hereditary property  $\mathcal{H}$  of *ordered* graphs there exists  $p = p_{\mathcal{H}}$  for which  $G \sim \mathbf{G}(n, p)$  satisfies  $d_1(G, \mathcal{H}) \geq \overline{d_{\mathcal{H}}} - o(1)$  with high probability?

As we show, the answer is in fact *negative*. Consider the ordered graph property  $\mathcal{H}$  defined as follows:  $G \in \mathcal{H}$  if and only if there do not exist vertices  $u_1 < u_2 \leq u_3 < u_4$  in  $G$  where  $u_1 u_2$  is a non-edge and  $u_3 u_4$  is an edge.  $\mathcal{H}$  is clearly a hereditary property, defined by a finite family of forbidden ordered subgraphs. In the full version [9], we prove that the typical distance of  $G \sim \mathbf{G}(n, p)$  from  $\mathcal{H}$  is no more than  $1/4 + o(1)$  (the maximum is asymptotically attained for  $p = 1/2$ ). In contrast, we show there exists a graph  $G$  satisfying  $d_1(G, \mathcal{H}) = 1/2 - o(1)$ , which is clearly the furthest possible up to the  $o(1)$  term (every graph  $G$  is  $1/2$ -close to either the complete or the empty graph, which are in  $\mathcal{H}$ ), and is substantially further than the typical distance of  $\mathbf{G}(n, p)$  for any choice of  $p$ . This shows that Theorem 7 *cannot be true* for the ordered setting.

However, the news are not all negative: We present a positive result in the ordered setting, which generalizes the unordered statement in some sense, and whose proof makes use of our ordered limit theory. While it is no longer true that  $\mathbf{G}(n, p)$  generates graphs that are asymptotically the furthest from  $\mathcal{H}$ , we show that a random graph generated according to a *consecutive stochastic block model* is approximately the furthest. A *stochastic block model* [1] with  $M$  blocks is a well-studied generalization of  $\mathbf{G}(n, p)$ , widely used in the study of community detection, clustering, and various other problems in mathematics and computer science. A stochastic block model is defined according to the following three parameters:  $n$ , the total number of vertices;  $(q_1, \dots, q_M)$ , a vector of probabilities that sum up to one; and a symmetric  $M \times M$  matrix of probabilities  $p_{ij}$ . A graph on  $n$  vertices is generated according to this model as follows. First, we assign each of the vertices independently<sup>9</sup> to one of  $M$  parts  $A_1, \dots, A_M$ , where the probability of any given vertex to fall in  $A_i$  is  $q_i$ . Then, for any  $(i, j) \in [M]^2$ , and any pair of disjoint vertices  $u \in A_i$  and  $v \in A_j$ , we add an edge between  $u$  and  $v$  with probability  $p_{ij}$ . By *consecutive*, we mean that all vertices assigned to  $A_i$  precede (in the vertex ordering) all vertices assigned to  $A_{i+1}$ , for any  $i \in [M - 1]$ . Our main result now is as follows.

<sup>9</sup> In some contexts, the stochastic block model is defined by determining the *exact* number of vertices in each  $A_i$  in advance, rather than assigning the vertices independently; all results here are also true for this alternative definition.

► **Theorem 8.** *Let  $\mathcal{H}$  be a hereditary property of simple ordered graphs and let  $\varepsilon > 0$ . There exists a consecutive stochastic block model with at most  $M = M_{\mathcal{H}}(\varepsilon)$  blocks with equal containment probabilities (i.e.,  $q_i = 1/M$  for any  $i \in [M]$ ), satisfying the following. A graph  $G$  on  $n$  vertices generated by this model satisfies  $d_1(G, \mathcal{H}) \geq \overline{d_{\mathcal{H}}} - \varepsilon$  with probability that tends to one as  $n \rightarrow \infty$ .*

The proof, given in the full version of this paper [9], is a good example of the power of the analytic perspective, combining our ordered limit theory with standard measure-theoretic tools and a few simple lemmas proved in [30].

## 1.4 Sampling and property testing

We finish by showing two additional applications of the ordered limit theory. These applications are somewhat more algorithmically oriented – concerning sampling and property testing – and illustrate the use of our theory for algorithmic purposes. The first of them is concerned with naturally estimable ordered graph parameters, defined as follows.

► **Definition 9** (naturally estimable parameter). *An ordered graph parameter  $f$  is naturally estimable if for every  $\varepsilon > 0$  and  $\delta > 0$  there is a positive integer  $k = k(\varepsilon, \delta) > 0$  satisfying the following. If  $G$  is an ordered graph with at least  $k$  nodes and  $G|_{\mathbf{k}}$  is the subgraph induced by a uniformly random ordered set of exactly  $k$  nodes of  $G$ , then*

$$\Pr_{G|_{\mathbf{k}}} [|f(G) - f(G|_{\mathbf{k}})| > \varepsilon] < \delta.$$

The following result provides an analytic characterization of ordered natural estimability, providing a method to study estimation problems on ordered graphs from the analytic perspective.

► **Theorem 10.** *Let  $f$  be a bounded simple ordered graph parameter. Then, the following are equivalent:*

1.  *$f$  is naturally estimable.*
2. *For every convergent sequence  $\{G_n\}_{n \in \mathbb{N}}$  of ordered simple graphs with  $|V(G_n)| \rightarrow \infty$ , the sequence of numbers  $\{f(G_n)\}_{n \in \mathbb{N}}$  is convergent.*
3. *There exists a functional  $\hat{f}(W)$  over  $\mathcal{W}$  that satisfies the following:*
  - a.  *$\hat{f}(W)$  is continuous with respect to  $d_{\Delta}$ .*
  - b. *For every  $\varepsilon > 0$ , there is  $k = k(\varepsilon)$  such that for every ordered graph  $G$  with  $|V(G)| \geq k$ , it holds that  $|\hat{f}(W_G) - f(G)| \leq \varepsilon$ .*

Our third application is a new analytic proof of the ordered graph removal lemma of [2], implying that every hereditary property of ordered graphs (and images over a fixed alphabet) is testable, with one-sided error, using a constant number of queries. (For the relevant definitions, see [2] and Definition 9 here.)

► **Theorem 11** ([2]). *Let  $\mathcal{H}$  be a hereditary property of simple ordered graphs, and fix  $\varepsilon, c > 0$ . Then there exists  $k = k(\mathcal{H}, \varepsilon, c)$  satisfying the following: For every ordered graph  $G$  on  $n \geq k$  vertices that is  $\varepsilon$ -far from  $\mathcal{H}$ , the probability that  $G|_{\mathbf{k}}$  does not satisfy  $\mathcal{H}$  is at least  $1 - c$ .*

Our proof of Theorem 11 utilizes the analytic tools developed in this work, and bypasses the need for many of the sophisticated combinatorial techniques from [2], resulting in an arguably cleaner proof.

## 1.5 Related work

The theory of graph limits has strong ties to the area of property testing, especially in the dense setting. Regularity lemmas for graphs, starting with the well-known regularity lemma of Szemerédi [34], later to be joined by the weaker (but more efficient) versions of Frieze and Kannan [22, 23] and the stronger variants of Alon et al. [3], among others, have been very influential in the development of property testing. For example, regularity was used to establish the testability of all hereditary properties in graphs [5], the relationship between the testability and estimability of graph parameters [19], and combinatorial characterizations of testability [4].

The analytic theory of convergence, built using the cut distance and its relation to the weak regularity lemma, has proved to be an interesting alternative perspective on these results. Indeed, the aforementioned results have equivalent analytic formulations, in which both the statement and the proof seem cleaner and more natural. A recent line of work has shown that many of the classical results in property testing of dense graphs can be extended to dense ordered graph-like structures, including vertex-ordered graphs and images. In [2], it was shown that the testability of hereditary properties extends to the ordered setting (see Theorem 11 above). Shortly after, in [8] it was proved that characterizations of testability in unordered graphs can be partially extended to similar characterizations in ordered graph-like structures, provided that the property at stake is sufficiently “well-behaved” in terms of order.

Graphons and their sparse analogues have various applications in different areas of mathematics, computer science, and even social sciences. The connections between graph limits and real-world large networks have been very actively investigated; see the survey of Borgs and Chayes [16]. Graph limits have applications in probability and data analysis [33]. Graphons were used to provide new analytic proofs of results in extremal graph theory; see Chapter 16 in [27]. Through the notion of free energy, graphons were also shown to be closely connected to the field of statistical physics [15]. We refer the reader to [27] for more details.

We remark that an independent work, by Frederik Garbe, Robert Hancock, Jan Hladky, and Maryam Sharifzadeh, investigates an alternative limit object for the ordered setting in the context of latin squares. See [24] for their findings, as well as connections between orderons and their limit object, called a latinon.

## 1.6 Organization

Due to space limitations, much of the technical content is missing from this version of the paper. Specifically, we only include here the following components. In Section 2, we present basic definitions for our ordered limit theory. Section 3 contains the main ingredients for the the proof that ordered graphs are dense in the space of orderons (some technical details are relegated to the full version). Section 4 presents the proof that the space of orderons is compact (Theorem 1). The reader is referred to [9] for a full version of this paper, including proofs of all results stated in this manuscript.

## 2 Preliminaries

In this section we formally describe some of the basic ingredients of our theory, including the limit object – the *orderon*, and several distance notions including the cut-norm for orderons (both unordered and ordered variants are presented), and the cut-shift distance. We then

show that the latter is a pseudo-metric for the space of orderons. This will later allow us to view the space of orderons as a metric space, by identifying orderons of cut-shift distance 0.

The measure used here is the Lebesgue measure, denoted by  $\lambda$ . We start with the formal definition of an orderon.

► **Definition 12 (orderon).** *An orderon is a measurable function  $W : ([0, 1]^2)^2 \rightarrow [0, 1]$  that is symmetric in the sense that  $W((x, a), (y, b)) = W((y, b), (x, a))$  for all  $(x, a), (y, b) \in [0, 1]^2$ . For the sake of brevity, we also denote  $W((x, a), (y, b))$  by  $W(v_1, v_2)$  for  $v_1, v_2 \in [0, 1]^2$ .*

We denote the set of all orderons by  $\mathcal{W}$ .

► **Definition 13 (measure-preserving bijection).** *A map  $g : [0, 1]^2 \rightarrow [0, 1]^2$  is measure preserving if the pre-image  $g^{-1}(X)$  is measurable for every measurable set  $X$  and  $\lambda(g^{-1}(X)) = \lambda(X)$ . A measure preserving bijection is a measure preserving map whose inverse map exists (and is also measure preserving).*

Let  $\mathcal{F}$  denote the collection of all measure preserving bijections from  $[0, 1]^2$  to itself. Given an orderon  $W \in \mathcal{W}$  and  $f \in \mathcal{F}$ , we define  $W^f$  as the unique orderon satisfying  $W^f((x, a), (y, b)) = W(f(x, a), f(y, b))$  for any  $x, a, y, b \in [0, 1]$ . Additionally, denote by  $\pi_1 : [0, 1]^2 \rightarrow [0, 1]$  the projection to the first coordinate, that is,  $\pi_1(x, a) = x$  for any  $(x, a) \in [0, 1]^2$ .

## 2.1 Cut-norm and ordered cut-norm

The definition of the (unordered) cut-norm for orderons is analogous to the corresponding definition for graphons.

► **Definition 14 (cut-norm).** *Given a symmetric measurable function  $W : ([0, 1]^2)^2 \rightarrow \mathbb{R}$ , we define the cut-norm of  $W$  as*

$$\|W\|_{\square} \stackrel{\text{def}}{=} \sup_{S, T \subseteq [0, 1]^2} \left| \int_{(x, a) \in S} \int_{(y, b) \in T} W((x, a), (y, b)) dx da dy db \right|.$$

As we are working with ordered objects, the following definition of *ordered cut-norm* will sometimes be of use (see the full version [9] for more details). Given  $v_1, v_2 \in [0, 1]^2$ , we write  $v_1 \leq v_2$  to denote that  $\pi_1(v_1) \leq \pi_1(v_2)$ . Let  $\mathbf{1}_E$  be the indicator function for the event  $E$ .

► **Definition 15 (ordered cut-norm).** *Let  $W : ([0, 1]^2)^2 \rightarrow \mathbb{R}$  be a symmetric measurable function. The ordered cut norm of  $W$  is defined as*

$$\|W\|_{\square'} = \sup_{S, T \subseteq [0, 1]^2} \left| \int_{(v_1, v_2) \in S \times T} W(v_1, v_2) \mathbf{1}_{v_1 \leq v_2} dv_1 dv_2 \right|.$$

We mention two important properties of the ordered-cut norm. The first is a standard smoothing lemma, and the second is a relation between the ordered cut-norm and the unordered cut-norm. The proof of both lemmas can be found in the full version [9].

► **Lemma 16.** *Let  $W \in \mathcal{W}$  and  $\mu, \nu : [0, 1]^2 \rightarrow [0, 1]$ . Then,*

$$\left| \int_{v_1, v_2} \mu(v_1) \nu(v_2) W(v_1, v_2) \mathbf{1}_{v_1 \leq v_2} dv_1 dv_2 \right| \leq \|W\|_{\square'}.$$

► **Lemma 17.** *Let  $W : ([0, 1]^2)^2 \rightarrow [-1, 1]$  be a symmetric measurable function. Then,*

$$\frac{\|W\|_{\square'}^2}{4} \leq \|W\|_{\square} \leq 2\|W\|_{\square'}.$$

## 2.2 The cut and shift distance

The next notion of distance is a central building block in this work. It can be viewed as a locality preserving variant of the unordered cut distance, which accounts for order changes resulting from applying a measure preserving function.

► **Definition 18.** *Given two orderons  $W, U \in \mathcal{W}$  we define the CS-distance (cut-norm+shift distance) as:*

$$d_{\Delta}(W, U) \stackrel{\text{def}}{=} \inf_{f \in \mathcal{F}} (\text{Shift}(f) + \|W - U^f\|_{\square}),$$

where  $\text{Shift}(f) \stackrel{\text{def}}{=} \sup_{x, a \in [0, 1]} |x - \pi_1(f(x, a))|$ .

► **Lemma 19.**  $d_{\Delta}$  is a pseudo-metric on the space of orderons.

For the proof, see the full version [9].

## 3 Block orderons and their density in $\mathcal{W}$

Here we show that weighted ordered graphs are dense in the space of orderons coupled with the cut-shift distance. To start, we have to define the orderon representation of a weighted ordered graph, called a *naive block orderon*. A naive  $n$ -block orderon is defined as follows.

► **Definition 20 (naive block orderon).** *Let  $m \in \mathbb{N}$ . For  $z \in (0, 1]$ , we denote  $Q_n(z) = \lceil nz \rceil$ ; we also set  $Q_n(0) = 1$ . An  $m$ -block naive orderon is a function  $W: ([0, 1]^2)^2 \rightarrow [0, 1]$  that can be written, for some weighted ordered graph  $G$  on  $n$  vertices, as*

$$W((x, a), (y, b)) = G(Q_n(x), Q_n(y)), \quad \forall x, a, y, b \in [0, 1].$$

Following the above definition, we denote by  $W_G$  the naive block orderon defined using  $G$ , and view  $W_G$  as the orderon “representing”  $G$  in  $\mathcal{W}$ . Similarly to the unordered setting, this representation is slightly ambiguous (but this will not affect us). Indeed, it is not hard to verify that two weighted ordered graphs  $F$  and  $G$  satisfy  $W_F = W_G$  if and only if both  $F$  and  $G$  are blowups of some weighted ordered graph  $H$ . Here, a weighted ordered graph  $G$  on  $nt$  vertices is a  $t$ -blowup of a weighted ordered graph  $H$  on  $n$  vertices if  $G(x, y) = H(\lceil x/t \rceil, \lceil y/t \rceil)$  for any  $x, y \in [nt]$ .

We call an orderon  $U \in \mathcal{W}$  a *step function with at most  $k$  steps* if there is a partition  $\mathcal{R} = \{S_1, \dots, S_k\}$  of  $[0, 1]^2$  such that  $U$  is constant on every  $S_i \times S_j$ .

► **Remark 21 (The name choices).** The definition of a step function in the space of orderons is the natural extension of a step function in graphons. Note that a naive block orderon is a special case of a step function, where the steps  $S_i$  are rectangular (this is why we call these “block orderons”). The “naive” prefix refers to the fact that we do not make use of the second coordinate in the partition.

For every  $W \in \mathcal{W}$  and every partition  $\mathcal{P} = \{S_1, \dots, S_k\}$  of  $[0, 1]^2$  into measurable sets, let  $W_{\mathcal{P}}: ([0, 1]^2)^2 \rightarrow [0, 1]$  denote the step function obtained from  $W$  by replacing its value at  $((x, a), (y, b)) \in S_i \times S_j$  by the average of  $W$  on  $S_i \times S_j$ . That is,

$$W_{\mathcal{P}}((x, a), (y, b)) = \frac{1}{\lambda(S_i)\lambda(S_j)} \int_{S_i \times S_j} W((x', a'), (y', b')) dx' da' dy' db',$$

Where  $i$  and  $j$  are the unique indices such that  $(x, a) \in S_i$  and  $(y, b) \in S_j$ , respectively.

The next lemma is an extension of the regularity lemma to the setting of Hilbert spaces.

## 42:12 Ordered Graph Limits and Their Applications

► **Lemma 22** ([29], Lemma 4.1). *Let  $\{\mathcal{K}_i\}_i$  be arbitrary non-empty subsets of a Hilbert space  $\mathcal{H}$ . Then, for every  $\varepsilon > 0$  and  $f \in \mathcal{H}$  there is an  $m \leq \lceil 1/\varepsilon^2 \rceil$  and there are  $f_i \in \mathcal{K}_i$  ( $1 \leq i \leq k$ ) and  $\gamma_1, \dots, \gamma_k \in \mathbb{R}$  such that for every  $g \in \mathcal{K}_{k+1}$*

$$|\langle g, f - (\gamma_1 f_1 + \dots + \gamma_k f_k) \rangle| \leq \varepsilon \|f\| \|g\|$$

The following is a direct consequence of Lemma 22.

► **Lemma 23.** *For every  $W \in \mathcal{W}$  and  $\varepsilon > 0$  there is a step function  $U \in \mathcal{W}$  with at most  $\lceil 2^{8/\varepsilon^2} \rceil$  steps such that*

$$\|W - U\|_{\square} \leq \varepsilon.$$

Similarly to the graphon case, the step function  $U$  might not be a stepping of  $W$ . However, it can be shown that these steppings are almost optimal.

► **Claim 24.** Let  $W \in \mathcal{W}$ , let  $U$  be a step function, and let  $\mathcal{P}$  denote the partition of  $[0, 1]^2$  into the steps of  $U$ . Then  $\|W - W_{\mathcal{P}}\|_{\square} \leq 2\|W - U\|_{\square}$ .

Using Lemma 23 and Claim 24 we can obtain the following lemma.

► **Lemma 25.** *For every function  $W \in \mathcal{W}$  and every  $\varepsilon > 0$ , there is a partition  $\mathcal{P}$  of  $[0, 1]^2$  into at most  $2^{\lceil 32/\varepsilon^2 \rceil}$  sets with positive measure such that  $\|W - W_{\mathcal{P}}\|_{\square} \leq \varepsilon$ .*

Using the above lemma, we can impose stronger requirements on our partition. In particular, we can show that there exists a partition of  $[0, 1]^2$  to sets of the same measure. Such a partition is referred to as an *equipartition*. Also, we say that a partition  $\mathcal{P}$  *refines*  $\mathcal{P}'$ , if  $\mathcal{P}$  can be obtained from  $\mathcal{P}'$  by splitting each  $P_j \in \mathcal{P}'$  into a finite number of sets (up to sets of measure 0).

► **Lemma 26.** *Fix some  $\varepsilon > 0$ . Let  $\mathcal{P}$  be an equipartition of  $[0, 1]^2$  into  $k$  sets, and fix  $q \geq 2k^2 \cdot 2^{162/\varepsilon^2}$  such that  $k$  divides  $q$ . Then, for any  $W \in \mathcal{W}$ , there exists an equipartition  $\mathcal{Q}$  that refines  $\mathcal{P}$  with  $q$  sets, such that  $\|W - W_{\mathcal{Q}}\|_{\square} \leq \frac{8\varepsilon}{9} + \frac{2}{k}$ .*

The next lemma is an (easier) variant of Lemma 26, in the sense that we refine two given partitions. However, the resulting partition will not be an equipartition.

► **Lemma 27.** *Fix some  $\varepsilon > 0$  and  $d \in \mathbb{N}$ . Let  $\mathcal{I}_d$  be an equipartition of  $[0, 1]^2$  into  $2^d$  sets,  $\mathcal{P}$  be a partition of  $[0, 1]^2$  into  $k$  sets, and fix  $q \geq 2(k \cdot 2^d)^2 \cdot 2^{162/\varepsilon^2}$  such that both  $k$  and  $2^d$  divide  $q$ . Then, for any  $W \in \mathcal{W}$ , there exists a partition  $\mathcal{Q}$  that refines both  $\mathcal{P}$  and  $\mathcal{I}_d$  with  $q$  sets, such that  $\|W - W_{\mathcal{Q}}\|_{\square} \leq \frac{8\varepsilon}{9} + \frac{2}{k \cdot 2^d}$ .*

**Proof.** Let  $\mathcal{P}' = \{P'_1, \dots, P'_{p'}\}$  be a partition of  $[0, 1]^2$  into  $p' \leq 2^{162/\varepsilon^2}$  sets such that  $\|W - W_{\mathcal{P}'}\|_{\square} \leq \frac{4\varepsilon}{9}$ , and let  $\mathcal{Q} = \{Q_1, \dots, Q_q\}$  be a common refinement of the three partitions  $\mathcal{P}$ ,  $\mathcal{P}'$  and  $\mathcal{I}_d$ . Note that we do not repartition further to get an equipartition. The rest of the proof is similar to the proof of Lemma 26. ◀

The following theorem shows that naive block orderons are a dense subset in  $\mathcal{W}$ .

► **Theorem 28.** *For every orderon  $W \in \mathcal{W}$  and every  $\varepsilon > 0$ , there exist a naive  $\frac{c}{\varepsilon^4} 2^{162/\varepsilon^2}$ -block orderon  $W'$  (for some constant  $c > 0$ ) such that*

$$d_{\Delta}(W, W') \leq \varepsilon.$$

**Proof.** Fix  $\varepsilon > 0$  and  $\gamma = \gamma(\varepsilon) > 0$ . We consider an interval equipartition  $J = \{J_1, \dots, J_{1/\gamma}\}$  of  $[0, 1]$  (namely, for each  $j \in [\frac{1}{\gamma} - 1]$ ,  $J_j = [(j-1) \cdot \gamma, j \cdot \gamma)$ , and for  $j = 1/\gamma$ ,  $J_j = [(j-1) \cdot \gamma, j \cdot \gamma]$ ). In addition, let  $\mathcal{P} = (J_i \times J_j \mid i, j \in [1/\gamma])$  be an equipartition of  $[0, 1]^2$ . By Lemma 26, there exists an equipartition  $\mathcal{Q}$  of  $[0, 1]^2$  of size  $q = \frac{2}{\gamma^4} 2^{162/\varepsilon^2}$  that refines  $\mathcal{P}$ , such that

$$\|W - W_{\mathcal{Q}}\|_{\square} \leq \frac{8\varepsilon}{9} + 2\gamma^2.$$

Next we construct a small shift measure preserving function  $f$  as follows. For every  $i \in [1/\gamma]$ , consider the collection of sets  $\{Q_k^i \mid k \in [\gamma q]\}$  in  $\mathcal{Q}$  such that

$$(J_i \times [0, 1]) \cap \mathcal{Q} = \{Q_k^i \mid k \in [\gamma q]\}.$$

For each  $k \in [\gamma q]$ , the function  $f$  maps  $Q_k^i$  to a rectangular set

$$\left[ (i-1)\gamma + \frac{(k-1)}{q}, (i-1)\gamma + \frac{k}{q} \right) \times [0, 1].$$

Finally, for every  $i, j \in [q]$  and every  $(x, a), (y, b) \in Q_i \times Q_j$ , we define

$$W'(f(x, a), f(y, b)) = W_{\mathcal{Q}}((x, a), (y, b))$$

Note that the resulting function  $W'$  obeys the definition of a naive  $q$ -block orderon and  $\text{Shift}(f) \leq \gamma$ . Therefore, setting  $\gamma = \varepsilon/100$ , we get that

$$d_{\Delta}(W, W') \leq \gamma + \frac{8\varepsilon}{9} + 2\gamma^2 \leq \varepsilon/100 + 8\varepsilon/9 + 2\varepsilon^2/100^2 \leq \varepsilon,$$

as desired. ◀

## 4 Compactness of the space of orderons

In this section we prove Theorem 1. We construct a metric space  $\widetilde{\mathcal{W}}$  from  $\mathcal{W}$  with respect to  $d_{\Delta}$ , by identifying  $W, U \in \mathcal{W}$  with  $d_{\Delta}(W, U) = 0$ . Let  $\widetilde{\mathcal{W}}$  be the image of  $\mathcal{W}$  under this identification. On  $\mathcal{W}$  the function  $d_{\Delta}$  is a distance function.

We start with some definitions and notations. Let  $(\Omega, \mathcal{M}, \lambda)$  be some probability space,  $\mathcal{P}_{\ell} = \{P_i^{(\ell)}\}_i$  a partition of  $\Omega$ , and let  $\beta(\mathcal{P}_{\ell} : \cdot) : \mathcal{P}_{\ell} \rightarrow [0, 1]$  be a function. For  $v \in \Omega$ , we slightly abuse notation and write  $\beta(\mathcal{P}_{\ell} : v)$  to denote  $\beta(\mathcal{P}_{\ell} : i)$  for  $v \in P_i^{(\ell)}$ . With this notation, observe that for every  $\ell$

$$\int_{v \in \Omega} \beta(\mathcal{P}_{\ell} : v) dv = \sum_{i \in [|\mathcal{P}_{\ell}|]} \lambda(P_i^{(\ell)}) \beta(\mathcal{P}_{\ell} : i). \quad (2)$$

The following two results serve as useful tools to prove convergence. The first result is known as the *martingale convergence theorem*, see e.g. Theorem A.12 in [27]. The second result is an application of the martingale convergence theorem, useful for our purposes.

► **Theorem 29** (see [27], Theorem A.12). *Let  $\{\mathbf{X}_i\}_{i \in \mathbb{N}}$  be a martingale satisfying  $\sup_n \mathbf{E}[|\mathbf{X}_n|] < \infty$ . Then  $\{\mathbf{X}_i\}_{i \in \mathbb{N}}$  is convergent with probability 1.*

► **Lemma 30.** *Let  $\{\mathcal{P}_{\ell}\}_{\ell}$  be a sequence of partitions of  $\Omega$  such that for every  $\ell$ ,  $\mathcal{P}_{\ell+1}$  refines  $\mathcal{P}_{\ell}$ . Assume that for every  $\ell$  and  $j \in [|\mathcal{P}_{\ell}|]$ , the functions  $\beta(\mathcal{P}_{\ell} : \cdot)$  satisfy*

$$\lambda(P_j^{(\ell)}) \beta(\mathcal{P}_{\ell} : j) = \sum_{i \in [|\mathcal{P}_{\ell+1}|]} \lambda(P_j^{(\ell)} \cap P_i^{(\ell+1)}) \beta(\mathcal{P}_{\ell+1} : i). \quad (3)$$

*Then, there is a measurable function  $\beta : \Omega \rightarrow [0, 1]$  such that  $\beta(v) = \lim_{\ell \rightarrow \infty} \beta(\mathcal{P}_{\ell} : v)$  for almost all  $v \in \Omega$ .*



**Proof.** Fix some  $\ell \in \mathbb{N}$ . Let  $\mathbf{X}$  be a uniformly distributed random variable in  $\Omega$ . Let  $\psi_\ell: \Omega \rightarrow [|\mathcal{P}_\ell|]$  be the function mapping each  $v \in \Omega$  to its corresponding part in  $\mathcal{P}_\ell$  and let  $\mathbf{Z}_\ell = \beta(\mathcal{P}_\ell: \mathbf{X})$ . We now show that the sequence  $(\mathbf{Z}_1, \mathbf{Z}_2, \dots)$  is a martingale. That is,  $\mathbf{E}_{\mathbf{X} \sim \Omega} [\mathbf{Z}_{\ell+1} \mid \mathbf{Z}_1, \dots, \mathbf{Z}_\ell] = \mathbf{Z}_\ell$ , for every  $\ell \in \mathbb{N}$ . Note that by the fact that  $\mathcal{P}_{\ell+1}$  refines  $\mathcal{P}_\ell$ ,  $\psi_\ell(\mathbf{X})$  determines  $\psi_i(\mathbf{X})$  for every  $i < \ell$ . By definition, the value  $\beta(\mathcal{P}_\ell: \mathbf{X})$  is completely determined by  $\psi_\ell(\mathbf{X})$ , and so it suffices to prove that  $\mathbf{Z}_\ell = \mathbf{E}_{\mathbf{X} \sim \Omega} [\mathbf{Z}_{\ell+1} \mid \psi_\ell(\mathbf{X})]$ . By the fact that for every  $j \in [|\mathcal{P}_\ell|]$  Equation (3) holds (and in particular holds for  $\psi_\ell(\mathbf{X})$ ), we can conclude that the sequence  $(\mathbf{Z}_1, \mathbf{Z}_2, \dots)$  is a martingale.

Since  $\mathbf{Z}_\ell$  is bounded, we can invoke the martingale convergence theorem (Theorem 29) and conclude that  $\lim_{\ell \rightarrow \infty} \mathbf{Z}_\ell$  exists with probability 1. That is,  $\beta(v) = \lim_{\ell \rightarrow \infty} \beta(\mathcal{P}_\ell: v)$  exists for almost all  $v \in \Omega$ .  $\blacktriangleleft$

► **Definition 31.** Fix some  $d \in \mathbb{N}$  and define  $\mathcal{I}_d = \{I_1^{(d)}, \dots, I_{2^d}^{(d)}\}$  so that for every  $t \in [2^d]$ ,  $I_t^{(d)} = \left[\frac{t-1}{2^d}, \frac{t}{2^d}\right) \times [0, 1]$ . We refer to this partition as the strip partition of order  $d$ .

The next lemma states that for any orderon  $W$  we can get a sequence of partitions  $\{\mathcal{P}_\ell\}_\ell$ , with several properties that will be useful later on.

► **Lemma 32.** For any orderon  $W \in \mathcal{W}$  and  $\ell \in \mathbb{N}$ , there is a sequence of partitions  $\{\mathcal{P}_\ell\}_\ell$  of  $[0, 1]^2$  with the following properties.

1.  $\mathcal{P}_\ell$  has  $g(\ell)$  many sets (for some monotone increasing  $g: \mathbb{N} \rightarrow \mathbb{N}$ ).
2. For every  $\ell$ ,  $\Gamma_\ell \stackrel{\text{def}}{=} \frac{g(\ell)}{g(\ell-1)} \in \mathbb{N}$ .
3. For every  $\ell' \geq \ell$ , the partition  $\mathcal{P}_{\ell'}$  refines both  $\mathcal{P}_\ell$  and the strip partition  $\mathcal{I}_{\ell'}$ . In particular, for every  $j \in [g(\ell-1)]$ ,

$$P_j^{(\ell-1)} = \bigcup_{j'=(j-1) \cdot \Gamma_\ell + 1}^{j \cdot \Gamma_\ell} P_{j'}^{(\ell)}.$$

4.  $W_\ell = (W)_{\mathcal{P}_\ell}$  satisfies  $\|W - W_\ell\|_\square \leq \frac{4}{g(\ell-1)2^\ell}$ .

**Proof.** We invoke Lemma 27 with the trivial partition  $\{[0, 1]^2\}$  and the strip partition  $\mathcal{I}_1$ , to get a partition  $\mathcal{P}_{n,1}$  with  $g(1)$  many sets such that  $\mathcal{P}_{n,1}$  refines  $\mathcal{I}_1$  and  $\|W_n - W_{n,1}\|_\square \leq 1$ . For  $\ell > 1$ , we invoke Lemma 27 with  $\mathcal{I}_\ell$  and  $\mathcal{P}_{n,\ell-1}$  to get a partition  $\mathcal{P}_{n,\ell}$  of size  $g(\ell) = (g(\ell-1) \cdot 2^\ell)^2 \cdot 2^{O(g(\ell-1)^2)}$  which refines both  $\mathcal{I}_\ell$  and  $\mathcal{P}_{n,\ell-1}$  such that  $\|W_n - W_{n,\ell}\|_\square \leq \frac{4}{g(\ell-1)2^\ell}$ . In order to take care of divisibility, we add empty (zero measure) sets in order to satisfy items (2) and (3).  $\blacktriangleleft$

Consider a sequence of orderons  $\{W_n\}_{n \in \mathbb{N}}$ . For every  $n \in \mathbb{N}$ , we use Lemma 32 to construct a sequence of functions  $\{W_{n,\ell}\}_\ell$  such that  $\|W_n - W_{n,\ell}\|_\square$  is small. For each  $\ell$ , we would like to approximate the shape of the limit partition resulting from taking  $n \rightarrow \infty$ . Inside each strip  $I_t^{(\ell)}$ , we consider the relative measure of the intersection of each set contained in  $I_t^{(\ell)}$ , with a finer strip partition  $\mathcal{I}_{\ell'}$ .

► **Definition 33 (shape function).** For fixed  $n \in \mathbb{N}$ , let  $\{\mathcal{P}_{n,\ell}\}_\ell$  be partitions of  $[0, 1]^2$  with the properties listed in Lemma 32. For every  $\ell' > \ell$  and  $I_{t'}^{(\ell')} \in \mathcal{I}_{\ell'}$ , we define  $\alpha_j^{(n,\ell)}(\mathcal{I}_{\ell'}: t') \stackrel{\text{def}}{=} 2^{\ell'} \cdot \lambda\left(P_j^{(n,\ell)} \cap I_{t'}^{(\ell')}\right)$  to be the relative volume of the set  $P_j^{(n,\ell)}$  in  $I_{t'}^{(\ell')}$ .

For any  $\ell' \geq \ell$  and  $I_{t'}^{(\ell')} \in \mathcal{I}_{\ell'}$ , by the compactness of  $[0, 1]$ , we can select a subsequence of  $\{W_n\}_{n \in \mathbb{N}}$  such that  $\alpha_j^{(n,\ell)}(\mathcal{I}_{\ell'}: t')$  converges for all  $j \in [g(\ell)]$  as  $n \rightarrow \infty$ . Let

$$\alpha_j^{(\ell)}(\mathcal{I}_{\ell'}: t') \stackrel{\text{def}}{=} \lim_{n \rightarrow \infty} \alpha_j^{(n,\ell)}(\mathcal{I}_{\ell'}: t').$$

Next we define the limit density function.

► **Definition 34** (density function). *For fixed  $n \in \mathbb{N}$ , let  $\{\mathcal{P}_{n,\ell}\}_\ell$  be partitions of  $[0, 1]^2$  with the properties listed in Lemma 32. We let  $\delta^{(n,\ell)}(\mathcal{P}_{n,\ell} \times \mathcal{P}_{n,\ell} : i, j) \stackrel{\text{def}}{=} W_{n,\ell}((x, a), (y, b))$  for  $(x, a) \in P_i^{(n,\ell)}$  and  $(y, b) \in P_j^{(n,\ell)}$ .*

*By the compactness of  $[0, 1]$ , we can select a subsequence of  $\{W_n\}_{n \in \mathbb{N}}$  such that  $\delta^{(n,\ell)}(\mathcal{P}_{n,\ell} \times \mathcal{P}_{n,\ell} : i, j)$  converge for all  $i, j \in [g(\ell)]$  as  $n \rightarrow \infty$ . Let*

$$\delta^{(\ell)}(i, j) \stackrel{\text{def}}{=} \lim_{n \rightarrow \infty} \delta^{(n,\ell)}(\mathcal{P}_{n,\ell} \times \mathcal{P}_{n,\ell} : i, j) .$$

The following lemma states that by taking increasingly refined strip partitions  $\mathcal{I}_{\ell'}$ , we obtain a limit shape function for each set contained in any strip of  $\mathcal{I}_\ell$ .

► **Lemma 35.** *For fixed  $\ell$  and  $j \in [g(\ell)]$ , there is a measurable function  $\alpha_j^{(\ell)} : [0, 1] \rightarrow [0, 1]$  such that  $\alpha_j^{(\ell)}(x) = \lim_{\ell' \rightarrow \infty} \alpha_j^{(\ell)}(\mathcal{I}_{\ell'} : x)$  for almost all  $x \in [0, 1]$ .*

**Proof.** Fix  $n, \ell$  and  $\ell' > \ell$ . For every  $j \in [g(\ell)]$ , by the definition of  $\alpha_j^{(n,\ell)}(\mathcal{I}_{\ell'} : t')$  and the strip partition  $\mathcal{I}_{\ell'}$

$$\lambda \left( I_{t'}^{(\ell')} \right) \cdot \alpha_j^{(n,\ell)}(\mathcal{I}_{\ell'} : t') = \lambda \left( P_j^{(n,\ell)} \cap I_{t'}^{(\ell')} \right) \quad \forall t' \in [2^\ell] .$$

On the other hand, since  $\mathcal{I}_{\ell'+1}$  refines  $\mathcal{I}_{\ell'}$ ,

$$\begin{aligned} \lambda \left( P_j^{(n,\ell)} \cap I_{t'}^{(\ell')} \right) &= \lambda \left( P_j^{(n,\ell)} \cap I_{2t'-1}^{(\ell'+1)} \right) + \lambda \left( P_j^{(n,\ell)} \cap I_{2t'}^{(\ell'+1)} \right) \\ &= \lambda \left( I_{2t'-1}^{(\ell'+1)} \right) \cdot \alpha_j^{(n,\ell)}(\mathcal{I}_{\ell'+1} : 2t' - 1) + \lambda \left( I_{2t'}^{(\ell'+1)} \right) \cdot \alpha_j^{(n,\ell)}(\mathcal{I}_{\ell'+1} : 2t') . \end{aligned}$$

Therefore, when  $n \rightarrow \infty$  we get that,

$$\lambda \left( I_{t'}^{(\ell')} \right) \cdot \alpha_j^{(\ell)}(\mathcal{I}_{\ell'} : t') = \lambda \left( I_{2t'-1}^{(\ell'+1)} \right) \cdot \alpha_j^{(\ell)}(\mathcal{I}_{\ell'+1} : 2t' - 1) + \lambda \left( I_{2t'}^{(\ell'+1)} \right) \cdot \alpha_j^{(\ell)}(\mathcal{I}_{\ell'+1} : 2t') ,$$

which is exactly the condition in Equation (3). By applying Lemma 30 with the sequence of strip partitions  $\{\mathcal{I}_{\ell'}\}_{\ell'}$  on  $\alpha_j^{(\ell)}$  the lemma follows. ◀

The next lemma asserts that the limit shape functions behave consistently.

► **Lemma 36.** *For every  $\ell$  and  $j \in [g(\ell - 1)]$ ,*

$$\alpha_j^{(\ell-1)}(x) = \sum_{j'=(j-1) \cdot \Gamma_\ell + 1}^{j \cdot \Gamma_\ell} \alpha_{j'}^{(\ell)}(x) ,$$

*for almost all  $x \in [0, 1]$ .*

**Proof.** Fix some  $n, \ell$  and  $\ell' > \ell$ . By the additivity of the Lebesgue measure,

$$\alpha_j^{(n,\ell-1)}(\mathcal{I}_{\ell'} : x) = \sum_{j'=(j-1) \cdot \Gamma_\ell + 1}^{j \cdot \Gamma_\ell} \alpha_{j'}^{(n,\ell)}(\mathcal{I}_{\ell'} : x) \quad \forall x \in [0, 1] .$$

By the fact that for every  $j \in [g(\ell - 1)]$  and  $x \in [0, 1]$  the sequence  $\left\{ \alpha_j^{(n,\ell-1)}(\mathcal{I}_{\ell'} : x) \right\}_n$  converges to  $\alpha_j^{(\ell-1)}(\mathcal{I}_{\ell'} : x)$  as  $n \rightarrow \infty$ , we get that

$$\alpha_j^{(\ell-1)}(\mathcal{I}_{\ell'} : x) = \sum_{j'=(j-1) \cdot \Gamma_\ell + 1}^{j \cdot \Gamma_\ell} \alpha_{j'}^{(\ell)}(\mathcal{I}_{\ell'} : x) \quad \forall x \in [0, 1] .$$

## 42:16 Ordered Graph Limits and Their Applications

By applying Lemma 35 on each  $j' \in [g(\ell)]$ , where  $\ell' \rightarrow \infty$ , we get that

$$\alpha_j^{(\ell-1)}(x) = \sum_{j'=(j-1) \cdot \Gamma_\ell + 1}^{j \cdot \Gamma_\ell} \alpha_{j'}^{(\ell)}(x),$$

for almost all  $x \in [0, 1]$ . ◀

Using the sequence of  $\{\alpha_j^{(\ell)}\}_j$  we define a limit partition  $\mathcal{A}_\ell = \{A_1^{(\ell)}, \dots, A_{g(\ell)}^{(\ell)}\}$  of  $[0, 1]^2$  as follows.

► **Definition 37** (limit partition). *For every  $\ell \in \mathbb{N}$ , let  $\mathcal{A}_\ell = \{A_1^{(\ell)}, \dots, A_{g(\ell)}^{(\ell)}\}$  be a partition of  $[0, 1]^2$  such that,*

$$A_j^{(\ell)} = \left\{ (x, a) : \sum_{i < j} \alpha_i^{(\ell)}(x) \leq a < \sum_{i \leq j} \alpha_i^{(\ell)}(x) \right\} \quad \forall j \in [g(\ell)].$$

► **Lemma 38.** *For any  $\ell$ , the partition  $\mathcal{A}_\ell$  has the following properties*

1.  $\mathcal{A}_\ell$  refines the strip partition  $\mathcal{I}_\ell$ .
2. The partition  $\mathcal{A}_\ell$  refines  $\mathcal{A}_{\ell-1}$ .
3. For every  $j \in [g(\ell)]$ ,  $\lambda(A_j^{(\ell)}) = \lim_{n \rightarrow \infty} \lambda(P_j^{(n, \ell)})$ .

**Proof.** The first item follows by the fact that each  $\alpha_j^{(\ell)}$  is non-zero inside only one strip.

By the definition of the sets  $A_j^{(\ell)}$  and Lemma 36 it follows that for each  $j \in [g(\ell-1)]$ ,

$$A_{j'}^{(\ell)} \subset A_j^{(\ell-1)} \quad \text{for all} \quad (j-1) \cdot \Gamma_\ell + 1 \leq j' \leq j \cdot \Gamma_\ell,$$

and therefore,

$$A_j^{(\ell-1)} = \bigcup_{j'=(j-1) \cdot \Gamma_\ell + 1}^{j \cdot \Gamma_\ell} A_{j'}^{(\ell)},$$

which shows the second item. To prove the third item of the lemma, note that for every  $n, \ell$  and  $\ell' > \ell$ ,

$$\begin{aligned} \lim_{n \rightarrow \infty} \lambda(P_j^{(n, \ell)}) &= \lim_{n \rightarrow \infty} \sum_{t' \in [2^{\ell'}]} 2^{-\ell'} \cdot \alpha_j^{(n, \ell)}(\mathcal{I}_{\ell'} : t') \\ &= \sum_{t' \in [2^{\ell'}]} 2^{-\ell'} \cdot \alpha_j^{(\ell)}(\mathcal{I}_{\ell'} : t') = \int_x \alpha_j^{(\ell)}(\mathcal{I}_{\ell'} : x) dx, \end{aligned}$$

where the last equality follows from Equation (2). Finally, by taking  $\ell' \rightarrow \infty$  and using Lemma 35, we get

$$\lim_{n \rightarrow \infty} \lambda(P_j^{(n, \ell)}) = \int_x \alpha_j^{(\ell)}(x) dx = \lambda(A_j^{(\ell)})$$

as desired. ◀

Using the definition of  $\delta^{(\ell)}$  and  $\mathcal{A}_\ell$ , we define a density function on the limit partition. For  $(x, a) \in A_i^{(\ell)}$  and  $(y, b) \in A_j^{(\ell)}$ , let

$$\delta(\mathcal{A}_\ell \times \mathcal{A}_\ell : (x, a), (y, b)) \stackrel{\text{def}}{=} \delta^{(\ell)}(i, j).$$

► **Lemma 39.** *For each  $\ell \in \mathbb{N}$  and  $i, j \in [g(\ell - 1)]$ ,*

$$\begin{aligned} \sum_{i'=(i-1) \cdot \Gamma_\ell + 1}^{i \cdot \Gamma_\ell} \sum_{j'=(j-1) \cdot \Gamma_\ell + 1}^{j \cdot \Gamma_\ell} \lambda(A_{i'}^{(\ell)}) \cdot \lambda(A_{j'}^{(\ell)}) \delta(\mathcal{A}_\ell \times \mathcal{A}_\ell : i', j') \\ = \lambda(A_i^{(\ell-1)}) \cdot \lambda(A_j^{(\ell-1)}) \delta(\mathcal{A}_{\ell-1} \times \mathcal{A}_{\ell-1} : i, j) . \end{aligned}$$

**Proof.** Fix  $n, \ell$  and  $i, j \in [g(\ell - 1)]$ . By the definition of the partitions  $\mathcal{P}_{n,\ell}$ ,  $\mathcal{P}_{n,\ell-1}$  and the density functions  $\delta^{(n,\ell)}$ ,  $\delta^{(n,\ell-1)}$

$$\begin{aligned} \sum_{i'=(i-1) \cdot \Gamma_\ell + 1}^{i \cdot \Gamma_\ell} \sum_{j'=(j-1) \cdot \Gamma_\ell + 1}^{j \cdot \Gamma_\ell} \lambda(P_{i'}^{(n,\ell)}) \cdot \lambda(P_{j'}^{(n,\ell)}) \delta^{(n,\ell)}(\mathcal{P}_{n,\ell} \times \mathcal{P}_{n,\ell} : i', j') \\ = \lambda(P_i^{(n,\ell-1)}) \cdot \lambda(P_j^{(n,\ell-1)}) \delta(\mathcal{P}_{n,\ell-1} \times \mathcal{P}_{n,\ell-1} : i, j) . \end{aligned}$$

By taking the limit as  $n \rightarrow \infty$  and using the third item of Lemma 38,

$$\begin{aligned} \sum_{i'=(i-1) \cdot \Gamma_\ell + 1}^{i \cdot \Gamma_\ell} \sum_{j'=(j-1) \cdot \Gamma_\ell + 1}^{j \cdot \Gamma_\ell} \lambda(A_{i'}^{(\ell)}) \cdot \lambda(A_{j'}^{(\ell)}) \delta(\mathcal{A}_\ell \times \mathcal{A}_\ell : i', j') \\ = \lambda(A_i^{(\ell-1)}) \cdot \lambda(A_j^{(\ell-1)}) \delta(\mathcal{A}_{\ell-1} \times \mathcal{A}_{\ell-1} : i, j) , \end{aligned}$$

which completes the proof. ◀

The next Lemma asserts that the natural density function of the limit partition is measurable. It follows directly from the combination of Lemma 30 and Lemma 39.

► **Lemma 40.** *There exists a measurable function  $\delta : ([0, 1]^2)^2 \rightarrow [0, 1]$  such that  $\delta((x, a), (y, a)) = \lim_{\ell \rightarrow \infty} \delta(\mathcal{A}_\ell \times \mathcal{A}_\ell : (x, a), (y, b))$  for almost all  $(x, a), (y, b) \in ([0, 1]^2)^2$ .*

Finally, we are ready to prove Theorem 1.

**Proof of Theorem 1.** We start by giving a high-level overview of the proof. Let  $\{W_n\}_{n \in \mathbb{N}}$  be a sequence of functions in  $\mathcal{W}$ . We show that there exists a subsequence that has a limit in  $\widetilde{\mathcal{W}}$ .

For every  $n \in \mathbb{N}$ , we use Lemma 32 to construct a sequence of functions  $\{W_{n,\ell}\}_\ell$  such that  $\|W_n - W_{n,\ell}\|_\square \leq \frac{4}{g(\ell-1)2^\ell}$ . Then, for every fixed  $\ell \in \mathbb{N}$ , we find a subsequence of  $\{W_{n,\ell}\}$  such that their corresponding  $\alpha_j^{(n,\ell)}$  and  $\delta^{(n,\ell)}(i, j)$  converge for all  $i, j \in [g(\ell)]$  (as  $n \rightarrow \infty$ ). For every  $\ell$ , we consider the partition  $\mathcal{A}_\ell$  (which by Definition 37, is determined by  $\{\alpha_j^{(\ell)}\}_j$ ) and  $\delta^{(\ell)}$ . Using  $\mathcal{A}_\ell$  and  $\delta^{(\ell)}$ , we can define the function  $U_\ell$ , such that  $W_{n,\ell} \rightarrow U_\ell$  almost everywhere as  $n \rightarrow \infty$ .

Given the sequence of functions  $\{U_\ell\}_\ell$ , we use Lemma 40 to show that  $\{U_\ell\}_\ell$  converges to some  $U$  almost everywhere as  $\ell \rightarrow \infty$  (where  $U$  is defined according the limit density function  $\delta$ ). Finally we show that for any fixed  $\varepsilon > 0$ , there is  $n_0(\varepsilon)$  such that for any  $n > n_0(\varepsilon)$ ,  $d_\Delta(W_n, U) \leq \varepsilon$ .

Fix some  $\varepsilon > 0$  and  $\xi(\varepsilon) > 0$  which will be determined later. Consider the sequence  $\{U_\ell\}_\ell$  which is defined by the partition  $\mathcal{A}_\ell$  and the density function  $\delta^{(\ell)}$ . By Lemma 40, the sequence  $\{U_\ell\}_\ell$  converges (as  $\ell \rightarrow \infty$ ) almost everywhere to  $U$ , which is defined by the limit density function  $\delta$ . Therefore, we can find some  $\ell > 1/\xi$  such that  $\|U_\ell - U\|_1 \leq \xi$ .

Fixing this  $\ell$ , we show that there is  $n_0$  such that  $d_\Delta(W_{n,\ell}, U_\ell) \leq 2^{-\ell} + 3\xi$  for all  $n > n_0$ . We shall do it in two steps by defining an interim function  $W'_{n,\ell}$  and using the triangle inequality.

## 42:18 Ordered Graph Limits and Their Applications

Recall that the function  $W_{n,\ell}$  is defined according to the partition  $\mathcal{P}_{n,\ell}$  and the density function  $\delta^{(n,\ell)}$ . Let  $W'_{n,\ell}$  be the function defined according to the partition  $\mathcal{A}_\ell$  and the density function  $\delta^{(n,\ell)}$ . That is, for every  $(x, a) \in A_i^{(\ell)}$  and  $(y, b) \in A_j^{(\ell)}$ ,  $W'_{n,\ell}((x, a), (y, b)) \stackrel{\text{def}}{=} \delta^{(n,\ell)}(\mathcal{P}_{n,\ell} \times \mathcal{P}_{n,\ell} : i, j)$ . By the third item of Lemma 38, for every  $j \in [g(\ell)]$ ,  $\lambda(A_j^{(\ell)}) = \lim_{n \rightarrow \infty} \lambda(P_j^{(n,\ell)})$ . Then, we can find  $n'_0(\ell)$  such that for all  $n > n'_0$ ,

$$\max\left(\lambda(A_j^{(\ell)}), \lambda(P_j^{(n,\ell)})\right) - \min\left(\lambda(A_j^{(\ell)}), \lambda(P_j^{(n,\ell)})\right) \leq \frac{\xi}{g(\ell)} \quad \forall j \in [g(\ell)]. \quad (4)$$

We define a measure preserving map  $f$  from  $W_{n,\ell}$  to  $W'_{n,\ell}$  as follows. For every strip  $I_t^{(\ell)} \in \mathcal{I}_\ell$ , we consider all the sets  $\{P_{j_1}^{(n,\ell)}, \dots, P_{j_t}^{(n,\ell)}\}$  in  $\mathcal{P}_{n,\ell}$  such that  $\bigcup_{j'=j_1}^{j_t} P_{j'}^{(n,\ell)} = I_t^{(\ell)}$ . Similarly, consider all the sets  $\{A_{j_1}^{(\ell)}, \dots, A_{j_t}^{(\ell)}\}$  in  $\mathcal{A}_\ell$  such that  $\bigcup_{j'=j_1}^{j_t} A_{j'}^{(\ell)} = I_t^{(\ell)}$ . For every  $j' \in \{j_1, \dots, j_t\}$ , we map an arbitrary subset  $S_{j'}^{(n,\ell)} \subseteq P_{j'}^{(n,\ell)}$  of measure  $\min\left(\lambda(A_{j'}^{(\ell)}), \lambda(P_{j'}^{(n,\ell)})\right)$  to an arbitrary subset (with the same measure) of  $A_{j'}^{(\ell)}$ . Next, we map  $I_t^{(\ell)} \setminus \bigcup_{j'=j_1}^{j_t} S_{j'}^{(n,\ell)}$  to  $I_t^{(\ell)} \setminus \bigcup_{j'=j_1}^{j_t} f(S_{j'}^{(n,\ell)})$ . Note that by (4) and the fact that  $W_{n,\ell}$  and  $W'_{n,\ell}$  have the same density function  $\delta^{(n,\ell)}$ , the functions  $W_{n,\ell}$  and  $W'_{n,\ell}$  disagree on a set of measure at most  $2\xi$ . Note that for every  $I_t^{(\ell)} \in \mathcal{I}_\ell$ , the function  $f$  maps sets from  $\mathcal{P}_{n,\ell}$  that are contained in  $I_t^{(\ell)}$  to sets in  $\mathcal{A}_\ell$  that are contained in  $I_t^{(\ell)}$ , and thus,  $\text{Shift}(f) \leq 2^{-\ell}$ . Therefore, for  $n > n'_0$ , we get that  $d_\Delta(W_{n,\ell}, W'_{n,\ell}) \leq 2^{-\ell} + 2\xi$ , and the first step is complete.

In the second step we bound  $d_\Delta(W'_{n,\ell}, U_\ell)$ . The two functions  $W'_{n,\ell}$  and  $U_\ell$  are defined on the same partition  $\mathcal{A}_\ell$ , however, their values are determined by the density functions  $\delta^{(n,\ell)}$  and  $\delta^{(\ell)}$  respectively. By the fact that  $\delta^{(n,\ell)}$  converges to  $\delta^{(\ell)}$  (as  $n \rightarrow \infty$ ), we can find  $n''_0(\ell)$  such that for all  $n > n''_0$ ,

$$\left|\delta^{(n,\ell)}(i, j) - \delta^{(\ell)}(i, j)\right| \leq \frac{\xi}{g(\ell)^2} \quad \forall i, j \in [g(\ell)].$$

Thus, for every  $n > n''_0$ , it holds that  $d_\Delta(W'_{n,\ell}, U_\ell) \leq \|W'_{n,\ell} - U_\ell\|_1 \leq \xi$ . By choosing  $n_0 = \max(n'_0, n''_0)$  we get that

$$d_\Delta(W_{n,\ell}, U_\ell) \leq d_\Delta(W_{n,\ell}, W'_{n,\ell}) + d_\Delta(W'_{n,\ell}, U_\ell) \leq 2^{-\ell} + 3\xi.$$

By putting everything together we get that for every  $n > n_0$

$$\begin{aligned} d_\Delta(W_n, U) &\leq d_\Delta(W_n, W_{n,\ell}) + d_\Delta(W_{n,\ell}, U_\ell) + d_\Delta(U_\ell, U) \\ &\leq \|W_n - W_{n,\ell}\|_\square + d_\Delta(W_{n,\ell}, U_\ell) + \|U_\ell - U\|_1 \\ &\leq O\left(\frac{1}{g(\ell-1)2^\ell}\right) + 2^{-\ell} + 3\xi + \xi. \end{aligned}$$

By our choice of  $\ell > 1/\xi$  we get that

$$d_\Delta(W_n, U) \leq 6\xi.$$

By choosing  $\xi = \varepsilon/6$  the theorem follows. ◀

## References

- 1 Emmanuel Abbe. Community detection and stochastic block models: Recent developments. *Journal of Machine Learning Research*, 18(177):1–86, 2018.
- 2 Noga Alon, Omri Ben-Eliezer, and Eldar Fischer. Testing hereditary properties of ordered graphs and matrices. In *Proceedings of the 58th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 848–858, 2017.
- 3 Noga Alon, Eldar Fischer, Michael Krivelevich, and Mario Szegedy. Efficient testing of large graphs. *Combinatorica*, 20(4):451–476, 2000.
- 4 Noga Alon, Eldar Fischer, Ilan Newman, and Asaf Shapira. A combinatorial characterization of the testable graph properties: It’s all about regularity. *SIAM Journal on Computing*, 39(1):143–167, 2009.
- 5 Noga Alon and Asaf Shapira. A characterization of the (natural) graph properties testable with one-sided error. *SIAM Journal on Computing*, 37(6):1703–1727, 2008.
- 6 Noga Alon and Uri Stav. What is the furthest graph from a hereditary property? *Random Structures & Algorithms*, 33(1):87–104, 2008.
- 7 Maria Axenovich and Ryan R. Martin. Multicolor and directed edit distance. *Journal of Combinatorics*, 2(4), 2011.
- 8 Omri Ben-Eliezer and Eldar Fischer. Earthmover resilience and testing in ordered structures. In *Proceedings of the 33rd Conference on Computational Complexity (CCC)*, pages 18:1–18:35, 2018.
- 9 Omri Ben-Eliezer, Eldar Fischer, Amit Levi, and Yuichi Yoshida. Limits of ordered graphs and their applications. Full version of this work. [arXiv:1811.02023](https://arxiv.org/abs/1811.02023).
- 10 Christian Borgs, Jennifer Chayes, and David Gamarnik. Convergent sequences of sparse graphs: A large deviations approach. *Random Structures & Algorithms*, 51(1):52–89, 2017.
- 11 Christian Borgs, Jennifer Chayes, and László Lovász. Moments of two-variable functions and the uniqueness of graph limits. *Geometric and Functional Analysis*, 19(6):1597–1619, 2010.
- 12 Christian Borgs, Jennifer Chayes, László Lovász, Vera T Sós, Balázs Szegedy, and Katalin Vesztergombi. Graph limits and parameter testing. In *Proceedings of the 38th ACM Symposium on the Theory of Computing (STOC)*, pages 261–270, 2006.
- 13 Christian Borgs, Jennifer Chayes, László Lovász, Vera T. Sós, and Katalin Vesztergombi. Counting graph homomorphisms. In *Topics in Discrete Mathematics*, pages 315–371. Springer Berlin Heidelberg, 2006.
- 14 Christian Borgs, Jennifer Chayes, László Lovász, Vera T. Sós, and Katalin Vesztergombi. Convergent sequences of dense graphs I: Subgraph frequencies, metric properties and testing. *Advances in Mathematics*, 219(6):1801–1851, 2008.
- 15 Christian Borgs, Jennifer Chayes, László Lovász, Vera T. Sós, and Katalin Vesztergombi. Convergent sequences of dense graphs II. multiway cuts and statistical physics. *Annals of Mathematics*, 176(1):151–219, 2012.
- 16 Christian Borgs and Jennifer T. Chayes. Graphons: A nonparametric method to model, estimate, and design algorithms for massive networks. *CoRR*, abs/1706.01143, 2017. [arXiv:1706.01143](https://arxiv.org/abs/1706.01143).
- 17 Persi Diaconis and Svante Janson. Graph limits and exchangeable random graphs. *Rendiconti di Matematica e delle sue Applicazioni. Serie VII*, 28(1):33–61, 2008.
- 18 Gábor Elek and Balázs Szegedy. A measure-theoretic approach to the theory of dense hypergraphs. *Advances in Mathematics*, 231(3):1731–1772, 2012.
- 19 Eldar Fischer and Ilan Newman. Testing versus estimation of graph properties. *SIAM Journal on Computing*, 37(2):482–501, 2007.
- 20 John M. Franks. *A (terse) introduction to Lebesgue integration*, volume 48 of *Student Mathematical Library*. American Mathematical Society, 2009.
- 21 Michael Freedman, László Lovász, and Alexander Schrijver. Reflection positivity, rank connectivity, and homomorphism of graphs. *Journal of the American Mathematical Society*, 20:37–51, 2007.

- 22 Alan Frieze and Ravi Kannan. The regularity lemma and approximation schemes for dense problems. In *Proceedings of the 37th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 12–20, 1996.
- 23 Alan Frieze and Ravi Kannan. Quick approximation to matrices and applications. *Combinatorica*, 19(2):175–220, 1999.
- 24 Frederik Garbe, Robert Hancock, Jan Hladký, and Maryam Sharifzadeh. Limits of latin squares. *arXiv preprint*, 2020. Extended abstract appeared in Eurocomb 2019 under the name *Theory of limits of sequences of Latin squares*. [arXiv:2010.07854](#).
- 25 Carlos Hoppen, Yoshiharu Kohayakawa, Carlos Gustavo Moreira, Balázs Ráth, and Rudini Menezes Sampaio. Limits of permutation sequences. *Journal of Combinatorial Theory, Series B*, 103(1):93–113, 2013.
- 26 Svante Janson. Poset limits and exchangeable random posets. *Combinatorica*, 31(5):529–563, 2011.
- 27 László Lovász. *Large networks and graph limits*, volume 60. American Mathematical Society, 2012.
- 28 László Lovász and Balázs Szegedy. Limits of dense graph sequences. *Journal of Combinatorial Theory, Series B*, 96(6):933–957, 2006.
- 29 László Lovász and Balázs Szegedy. Szemerédi’s lemma for the analyst. *Geometric And Functional Analysis*, 17(1):252–270, 2007.
- 30 László Lovász and Balázs Szegedy. Testing properties of graphs and functions. *Israel Journal of Mathematics*, 178(1):113–156, 2010.
- 31 Ryan R. Martin. The edit distance in graphs: Methods, results, and generalizations. In Andrew Beveridge, Jerrold R. Griggs, Leslie Hogben, Gregg Musiker, and Prasad Tetali, editors, *Recent Trends in Combinatorics*, pages 31–62. Springer International Publishing, 2016.
- 32 Ryan R. Martin and Maria Axenovich. Avoiding patterns in matrices via a small number of changes. *SIAM Journal of Discrete Mathematics*, 20(1):49–54, 2006.
- 33 Peter Orbanz and Daniel M. Roy. Bayesian models of graphs, arrays and other exchangeable random structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(2):437–461, 2015.
- 34 Endre Szemerédi. Regular partitions of graphs. In *Problèmes combinatoires et théorie des graphes. Colloq. Internat. CNRS*, volume 260, pages 399–401, 1976.
- 35 Yuichi Yoshida. Gowers norm, function limits, and parameter estimation. In *Proceedings of the 27th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1391–1406, 2016.
- 36 Yufei Zhao. Hypergraph limits: A regularity approach. *Random Structures & Algorithms*, 47(2):205–226, 2015.



# Error Correcting Codes for Uncompressed Messages

Ofer Grossman

MIT, Cambridge, MA, USA  
ogrossma@mit.edu

Justin Holmgren

NTT Research, Palo Alto, CA, USA  
justin.holmgren@ntt-research.com

---

## Abstract

Most types of messages we transmit (e.g., video, audio, images, text) are not fully compressed, since they do not have known efficient and information theoretically optimal compression algorithms. When transmitting such messages, standard error correcting codes fail to take advantage of the fact that messages are not fully compressed.

We show that in this setting, it is sub-optimal to use standard error correction. We consider a model where there is a set of “valid messages” which the sender may send that may not be efficiently compressible, but where it is possible for the receiver to recognize valid messages. In this model, we construct a (probabilistic) encoding procedure that achieves better tradeoffs between data rates and error-resilience (compared to just applying a standard error correcting code).

Additionally, our techniques yield improved efficiently decodable (probabilistic) codes for fully compressed messages (the standard setting where the set of valid messages is all binary strings) in the high-rate regime.

**2012 ACM Subject Classification** Theory of computation → Error-correcting codes

**Keywords and phrases** Coding Theory, List Decoding

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.43

**Related Version** The full version of this paper is available at [11], <https://eccc.weizmann.ac.il/report/2020/038/>.

**Supplementary Material** The code used to generate the figures and tables in this paper can be found at [https://github.com/jnhnum1/contextual\\_codes](https://github.com/jnhnum1/contextual_codes).

**Funding** Work done in part while generously hosted by the Simons Institute.

*Ofer Grossman:* Supported by a Fannie and John Hertz Foundation Fellowship and a NSF GRFP fellowship.

*Justin Holmgren:* Work done in part while at Princeton University, supported by the Simons Collaboration on Algorithms and Geometry and National Science Foundation grant No. CCF-1714779.

**Acknowledgements** We thank Elad Haramaty and Madhu Sudan for many very fruitful discussions in the early stages of this work. We also thank Venkat Guruswami and Lijie Chen for helpful discussions, and Shafi Goldwasser for her encouragement and her feedback on an earlier version of this work. We also thank anonymous reviewers for many useful comments on earlier versions of this work.

## 1 Introduction

If Alice wishes to send a message  $m$  to Bob, she might first compress it as well as she can. In this work, we focus on *lossless* compression, meaning that Bob must recover  $m$  *exactly*. There are many types of data (e.g. images, audio, video, text) that we do not know how



© Ofer Grossman and Justin Holmgren;

licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 43; pp. 43:1–43:18

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

to compress and decompress efficiently and information-theoretically optimally. For such messages, this compression step will result in a longer-than-optimal message. For example, the best efficient compression scheme may result in a  $b$ -bit long compressed message, whereas an information theoretically optimal compression scheme might be able to obtain  $0.5b$  bits.

After compressing, Alice can apply an error correcting code to the compressed message, ensuring that Bob can recover the message in the presence of corruptions. Suppose Alice wishes to have her message be resilient against up to 5% worst-case errors. Then, the best known construction of a code with efficient unique decoding and public randomness will result in a total of approximately  $3.64b$  bits sent to Bob (using bounds implicit in [22, 17]).

We show that in this setting it is sub-optimal to treat compression and error correction as two orthogonal concerns. We instead address both compression and error correction simultaneously, constructing an error correcting code that exploits the fact that messages are not fully compressed. This allows Alice to send only fewer bits to Bob with the same error resilience. For example, with the above parameters Alice can send  $2.24b$  bits.

## 1.1 Contextually Unique Decoding

We define a new notion, *contextually unique decoding*, that formalizes the idea of encoding a message that is not fully compressed. Roughly speaking, we let  $S \subseteq \{0, 1\}^k$  denote a set of “valid messages” that Alice may send. Suppose, for example, that Alice is sending English text of a certain size to Bob. Then we think of  $S$  as the set of all “reasonable” texts Alice can send. For example, “meet me at 5pm” (when translated to binary) is in  $S$ . However, “wef ojip447oll” is not in  $S$ . We assume that Bob has oracle access to  $S$  – he has the power to determine whether a message is reasonable or not. Because most strings do not look like reasonable texts, we see that  $S$  is pretty small<sup>1</sup>. We now wish to say that whenever Alice encodes an element  $m$  of  $S$ , Bob will be able to recover  $m$ .

Motivated by this, we say that a family of codes  $\{C_i : \{0, 1\}^k \rightarrow \{0, 1\}^n\}$  is *contextually uniquely decodable* if there is a decoding algorithm  $D$  such that for any sufficiently small set of messages  $S \subseteq \{0, 1\}^k$ , it holds w.h.p. for a random  $i$  that for all  $m \in S$ , the algorithm  $D$  (with oracle access to  $S$ ) can recover  $m$  given  $i$  and an adversarially corrupted  $C_i(m)$  (where the adversary may depend on  $i$ ).

Alice may have partially compressed the text she wishes to send (we assume she cannot fully compress it, since we don’t know any efficient practical information theoretically optimal compression schemes for text). In this case, a message is in  $S$  if it looks like reasonable text once it is decompressed.

A code with contextually unique decoding would (assuming public randomness) allow Alice to send a message to Bob, so that he can recover Alice’s message even in the presence of corruptions.

Formally, we define:

► **Definition 1** ( $\delta$ -Hamming Adversary). *A  $\delta$ -Hamming adversary is a function  $\mathcal{A} : \{0, 1\}^n \rightarrow \{0, 1\}^n$  such that for all  $c \in \{0, 1\}^n$ , the Hamming distance between  $c$  and  $\mathcal{A}(c)$  is at most  $\delta n$ .*

<sup>1</sup> Notice that if  $S$  is of size  $2^{k'}$ , then information theoretically it would be possible for Alice to compress the message to  $k'$  bits, and then apply an error correcting code.

► **Definition 2** (Contextually Unique Decoder). *An oracle algorithm  $D$  is an  $(r, \delta, \epsilon, \tau)$ -contextually unique decoder for a family of probabilistic codes  $\{C_i : \{0, 1\}^k \xrightarrow{\$} \{0, 1\}^n\}_{i \in \mathcal{I}}$  if for all sets  $S \subseteq \{0, 1\}^k$  with  $|S| \leq 2^{rn}$ , it holds with probability at least  $1 - \epsilon$  over the choice of  $i \leftarrow \mathcal{I}$  that for all messages  $m \in S$  and all  $\delta$ -Hamming adversaries  $\mathcal{A}$  (that may depend on  $i$ ),*

$$\Pr_{c \leftarrow C_i(m)} [D^S(i, \mathcal{A}(c)) \neq m] \leq \tau.$$

We emphasize the order of quantifiers in the definition. We use randomness in two different ways. First, randomness is used to pick a code from the family  $\{C_i\}$ . This choice of randomness is agreed upon by all parties ahead of time and is publicly known (to the sender, receiver, and the adversary), and must work for all messages. Randomness is then also used by the sender (Alice) when encoding. That is, even after fixing the message  $m$  and the choice of  $C_i$ , the encoding of a message  $m$  using  $C_i$  depends on the encoder's randomness (we use the notation  $C_i : \{0, 1\}^k \xrightarrow{\$} \{0, 1\}^n$  to denote that  $C_i$  is a function taking an input from  $\{0, 1\}^k$ , along with some randomness, and outputs an element of  $\{0, 1\}^n$ , which may depend on the randomness). The decoder only needs to know the randomness used in picking  $C_i$ , and not the randomness used by the encoder in *evaluating*  $C_i$ .

## 1.2 Main Result and Construction Overview

We first overview our construction of contextually unique decodable codes, and then we formally describe our main result (Theorem 3).

### 1.2.1 Construction Overview

#### The Main Idea

In the standard model of error correcting codes, we have a code  $C$ , which we use to encode a message  $m$  as  $C(m)$ . Then, even when an adversary may corrupt a bounded number of entries of  $C(m)$ , it is still possible to recover  $m$ . This is called unique decoding. List decoding [6, 28] is a generalization of unique decoding where instead of recovering  $m$ , the decoding algorithm outputs a short (polynomial sized) list  $m_1, m_2, \dots, m_\ell$  such that the real message  $m$  is in the list. This relaxation makes it possible to handle more errors.

The key in our construction is to have Alice send a coded version of the message  $m$  with good list decoding properties. Then, the goal will be that when Bob list-decodes Alice's message, only one of the elements in Bob's list will be a "valid message" (that is, only one element of the list will be in  $S$ ). Then, the hope is that Bob can correct errors up to the list decoding radius, instead of the unique decoding radius.

So for example, Alice might encode the message "call me at 4pm", and after an adversary adds some errors, and Bob decodes, he will have a list of messages. Ideally, the list will look something like "kwjlewf 6oahzm", "aowi2ifmlpzo", "wef ojip447oll", "call me at 4pm", and "5ncbzmap89pqq". From this list, it will be easy for Bob to infer that the message Alice sent was "call me at 4pm" (formally, he will use his oracle access to  $S$  to check which of the strings are in  $S$ , and we hope that only one will be in the set  $S$  of valid messages). Note, however, that if Bob's list contains more than one valid message – for example, if the list of messages is "call me at 7pm", "my phone broke", "call me at 4pm", and "come to my office" – then it will not be possible for Bob to determine which was the intended message (formally, this situation corresponds to Bob's list containing more than one element in  $S$ ).

The technical focus of our constructions is ensuring that within the set of candidate messages provided by a list-decoding algorithm, with high probability only one message will be valid. Our main theorem (Theorem 3) can indeed be viewed as a transformation from a list-decodable code to a contextually-uniquely decodable code.

### Randomizing the message space

Let  $S$  be the set of valid messages. To ensure that only one elements of Bob's list is in  $S$ , the idea is to randomize the message space. If the messages in Bob's list are more or less random (other than Alice's intended message), it is unlikely that more than one of the messages will be in  $S$  (we assume that  $|S|$  is small relative to the entire message space  $\{0, 1\}^k$ ). So ideally, what we would want to do is pick a random permutation  $\pi$  of  $\{0, 1\}^k$  (which is the set of all possible messages, including those not in  $S$ ) using public randomness, and then use an error correcting code  $C$  with good list decoding parameters on  $\pi(m)$ . Then, the set of messages that the adversary can cause to be in Bob's list will be a random small subset of  $\{0, 1\}^k$ , which, because  $S$  is small, will likely not intersect  $S$ .

There are some issues with the approach described above. One issue is that picking a random permutation of  $\{0, 1\}^k$  requires a number of bits exponential in  $k$ , but we want Alice and Bob to be efficient. This issue can be solved by using pairwise independence. Roughly speaking, one can see why pairwise independence is enough as follows. The choice of  $\pi$  is bad if there are two messages  $m_1$  and  $m_2$  in  $S$  such that  $C(\pi(m_1))$  is close to  $C(\pi(m_2))$ . This causes the adversary to be able to corrupt few entries of an encoding of  $m_1$  and cause it to be close to an encoding of  $m_2$ . One can see that the probability  $C(\pi(m_1))$  is close to  $C(\pi(m_2))$  is the same for a random  $\pi$  and a  $\pi$  chosen from a pairwise independent family, since it depends on only two evaluations of  $\pi$ .

Another issue with the construction as described above is that the probability that there exist  $m_1, m_2 \in S$  with  $C(\pi(m_1))$  close to  $C(\pi(m_2))$  is not that low (it is  $2^{-cn}$ , for some  $c$ . Ideally, we would like it to be  $2^{-\omega(n)}$ , so we can apply a union bound over all of  $S$  and not worry about the value of  $c$ ). We alter the construction by instead of picking a single  $\pi$ , picking a collection  $\pi_1, \pi_2, \dots, \pi_N$  which will be agreed on using public randomness. Then, the encoder (Alice) will pick a random  $j \in [N]$ , and use  $C(\pi_j(m))$  as the message sent to Bob. To decode, Bob will decode  $C$  to get a list  $L$ , and then for each  $j \in [N]$  and for each  $x \in L$  he will check if  $\pi_j^{-1}(x) \in S$ , and with high probability only one such pair  $(x, j)$  will satisfy  $\pi_j^{-1}(x) \in S$ . Then Bob will know that the message  $m$  that Alice sent is  $\pi_j^{-1}(x)$ .

We now outline how we show that Bob's list with high probability indeed contains only one element in  $S$ . Consider the probability that for a certain  $m$  in  $S$ , we have  $C(\pi(m))$  close to some  $C(\pi(m'))$ . Call this probability  $p$ . Then, the probability that for most  $j$ , we have  $C(\pi_j(m))$  close to some  $C(\pi_j(m'))$  will be approximately  $p^{\Omega(N)}$  (by a Chernoff bound), which is much smaller than  $p$ . This allows us to apply a union bound over all messages in  $S$  without losing anything significant.

### Amplifying the success probability (Section 4)

The construction described above works, but it has a downside that there is inverse polynomial probability of error. That is, with probability inversely polynomial in the message lengths, Bob may be unable to recover the message Alice sent, since with probability approximately  $\frac{1}{N}$  Alice may pick a bad choice of  $i$ .

Ideally, we would want to succeed with all but negligible probability. One approach is to set  $N$  to be superpolynomial. The problem with this is that now Bob will not be able to efficiently decode, since his decoding algorithm requires trying every one of the  $N$

permutations. To fix this, instead of Alice sending Bob  $C(\pi_i(m))$ , she will send him  $C(\pi_i(m), i)$  (we apply  $C$  to the string which is the concatenation of  $\pi_i(m)$  and  $i$ ). Now, when Bob list decodes  $C$ , he will get a list of the form  $(x_1, i_1), (x_2, i_2), \dots, (x_\ell, i_\ell)$ . Now, he can check if  $\pi_{i_1}^{-1}(x_1) \in S$ , or if  $\pi_{i_2}^{-1}(x_2) \in S$ , and so on. Crucially, we see that for each element  $(x_j, i_j)$  in the list, Bob needs to try only a single permutation (namely  $i_j$ ), instead of all permutations. This allows Bob to remain efficient even when there are superpolynomially many permutations for Alice to pick from.

This leads to a new issue, since one needs to agree on a superpolynomially sized family of permutations sampled from a family of pairwise independent permutations. Also, we want this family to be efficiently sampleable, and for each element to have a succinct description. It turns out that this can be solved using  $k$ -wise independence (and  $k$ -wise  $\epsilon$ -dependence). This part is more technical, and we refer the reader to the body of the paper for details.

### 1.2.2 The main theorem

Here we formally describe the main theorem. We say that an  $n$ -bit code is **combinatorially**  $(\rho, \lambda)$ -list decodable if for any  $y \in \{0, 1\}^n$ , there are at most  $\approx 2^{\lambda n}$  codewords within relative Hamming distance  $\rho$  of  $y$ . We are also interested in the asymptotic computational efficiency of encoding and decoding procedures, so we consider ensembles of codes  $\{C_n : \{0, 1\}^{k_n} \rightarrow \{0, 1\}^n\}_{n \in \mathbb{Z}^+}$ . We will restrict our attention to ensembles where  $r = \lim_{n \rightarrow \infty} \frac{k_n}{n}$  exists, and we call  $r$  the **rate** of the ensemble. We say that  $\{C_n\}$  is **efficiently**  $\rho$ -list-decodable if there is a polynomial-time algorithm that on input  $y \in \{0, 1\}^n$ , outputs all codewords of  $C_n$  that are within relative Hamming distance  $\rho$  of  $y$ .

► **Theorem 3** (Simplified Main Theorem). *Suppose that  $\{C'_n : \{0, 1\}^{k'_n} \rightarrow \{0, 1\}^n\}_{n \in \mathbb{Z}^+}$  is a rate- $r'$  ensemble of (deterministic) codes that is efficiently  $\rho$ -list-decodable. Suppose also that  $\{C'_n\}$  is combinatorially  $(2\rho, \lambda)$ -list decodable.*

*Then for some negligible function  $\epsilon(n)$ , there is a rate- $r'$  ensemble of probabilistic codes  $\{C_n\}$  such that  $C_n$  has a polynomial-time  $(r' - \lambda - o(1), \rho, \epsilon, \epsilon)$ -contextually unique decoder.*

So, suppose we wish to construct contextually unique codes where the message can be recovered when there are up to 0.1 fraction of corruptions. So we have  $\rho = 0.1$ . We now wish to find an  $r'$  and  $\lambda$  which maximize  $r' - \lambda$  such that there are deterministic codes which are of rate  $r'$ , and are combinatorially  $(2\rho, \lambda)$ -list decodable (we also have make sure the codes are *efficiently*  $\rho$ -list decodable).

Once fixing  $\rho$ , the tradeoff here is between  $r'$  and  $\lambda$ . The best contextually unique codes will have high rates  $r' - \lambda - o(1)$ , so we want  $\lambda$  to be small, and  $r'$  to be large. However, the codes  $\{C'_n\}$  must be combinatorially  $(2\rho, \lambda)$ -list decodable. So, as we increase  $r'$ , the lowest possible value of  $\lambda$  decreases.

We give some examples of parameter settings to Theorem 3 in Table 1.

### 1.3 Improvements for standard randomized setting

An important special case of contextually unique decoding is obtained by fixing  $S = \{0, 1\}^{k_n}$ , viewed as a subset of  $\{0, 1\}^{k'_n}$  for  $k'_n > k_n$  by zero-padding. In this case, a contextually uniquely decodable code is quite similar to a standard error-correcting code – the main difference is in the use of randomness both in generating the code and encoding messages. Perhaps surprisingly, we obtain better parameters in the high-rate regime than any other known efficiently decodable code (including probabilistic constructions with public randomness).

■ **Table 1** This table shows some example values of what rates can be achieved with our main theorem (Theorem 3) together with the best of the Blokh-Zyablov (Fact 6) and Thommesen-Rudra bounds. If  $|S| = 2^{s \cdot k}$ , and  $S \subseteq \{0, 1\}^k$ , we say that the *sparsity* of  $S$  is  $s$ . So, for example, if  $S$  is of size  $2^{.5k}$ , and there are 3% fraction of errors, we see that we can achieve rate .574 (and so Alice’s message size would be  $k/.574$ , or approximately  $1.74k$ ). The best Alice would be able to do without contextually unique decoding would be rate .396, which corresponds to over  $2.52k$  bits sent (this can be seen by looking at the sparsity 1 row, which corresponds to using standard unique decoding, since in this case  $S$  is the whole message space  $\{0, 1\}^k$ ).

Errors \ Sparsity	0.01	0.02	0.03	0.05	0.1	0.2
1	0.661	0.504	0.396	0.275	0.142	0.028
0.9	0.778	0.591	0.451	0.277	0.142	0.030
0.75	0.778	0.661	0.574	0.332	0.142	0.034
0.5	0.778	0.661	0.574	0.446	0.142	0.038
0.25	0.778	0.661	0.574	0.446	0.142	0.040
0.1	0.778	0.661	0.574	0.446	0.178	0.041
0.05	0.778	0.661	0.574	0.446	0.202	0.041
0.01	0.778	0.661	0.574	0.446	0.235	0.041

## Discussion

If there is a (deterministic) code that can be *efficiently* list decoded up to  $\rho$  errors, and *combinatorially* uniquely decoded up to  $\rho'$  errors, then it is possible to efficiently uniquely decode up to  $\min(\rho, \rho')$  errors. This can be done by simply list decoding, and then going over every element in the list to determine which of the elements, when encoded, is closest to the received message. So, in short, it is not hard to see that good efficient list decoding and good combinatorial unique decoding implies good efficient unique decoding (this idea is, roughly speaking, what gives the green line (TR bound) in Figure 1).

Our corollary can be viewed as a strengthening of this result. We show that rather than requiring good efficient list decoding and good combinatorial unique decoding, we can use codes with good efficient list decoding and good combinatorial *list* decoding. The idea, roughly speaking, is to use our efficient list decoding algorithm to obtain a list of candidate messages, and we use our main theorem on contextually unique decoding to ensure that only one of these messages will be a “valid” message. Then, we can go through each candidate in the list, and pick the one which is a valid message.

► **Corollary 4** (Simplified Main Corollary). *Suppose that  $\{\tilde{C}^{(n)} : \{0, 1\}^{k'_n} \rightarrow \{0, 1\}^n\}_{n \in \mathbb{Z}^+}$  is a rate- $r'$  ensemble of (deterministic) codes that is efficiently  $\rho$ -list-decodable. Suppose also that  $\{\tilde{C}^{(n)}\}$  is combinatorially  $(2\rho, \lambda)$ -list decodable.*

*Then there exists an ensemble  $\{C^{(n)}\}_{n \in \mathbb{Z}^+}$ , where  $C^{(n)} = \{C_i : \{0, 1\}^{k_n} \xrightarrow{\$} \{0, 1\}^n\}_{i \in \mathcal{I}^{(n)}}$  is a family of probabilistic codes, such that:*

1.  $\lim_{n \rightarrow \infty} \frac{k_n}{n} = r' - \lambda$ , and
2. *There are poly( $n$ )-time algorithms to:*
  - *Sample from  $\mathcal{I}^{(n)}$  given  $1^n$ .*
  - *Probabilistically encode  $C_i(m)$  given  $i$  and  $m$ ,*
  - *Decode  $\rho$ -corrupted codewords of  $C_i$ . That is, there is a deterministic poly( $n$ )-time algorithm  $D$  and a negligible function  $\epsilon(n)$  such that with probability at least  $1 - \epsilon(n)$  over the choice of  $i \leftarrow \mathcal{I}^{(n)}$ , it holds for all messages  $m \in \{0, 1\}^{k_n}$  and all  $\rho$ -Hamming adversaries  $\mathcal{A}$ , that*

$$\Pr_{c \leftarrow C_i(m)} [D(i, \mathcal{A}(c)) \neq m] \leq \epsilon(n).$$

► **Remark 5.** An interesting weakening of the conclusion of Corollary 4 is that for sufficiently large  $n$ , there exists  $i \in \mathcal{I}^{(n)}$  such that for all messages  $m \in \{0, 1\}^{k_n}$ , there exists randomness  $s$  such that for all  $c' \approx_\delta C_i(m; s)$ , it holds that  $D(i, c') = m$ . In other words,  $D(i, \cdot)$  is a polynomial-size error-correcting circuit for an (inefficiently computable and non-explicit) *deterministic* code.

We compare the conclusion of Corollary 4 to what is known for standard (deterministic) binary codes.

To our knowledge, the best known rate vs. error tolerance tradeoff for efficiently decodable and deterministic binary codes is given by the Blokh-Zyablov (BZ) bound [2], and is attained by multi-level concatenated codes (see Fact 6). With probabilistically constructed codes, it is possible to do better for sufficiently low rates. It is known that for rates below about 0.02, the concatenation of a folded Reed-Solomon code with random linear codes simultaneously achieves high distance (matching the GV bound) [26] and efficient list-decodability for a larger number of errors [22]. This implies an efficient unique decoding procedure (by list decoding and then taking the candidate that is closest to the received word). While not made explicit in previous work, the same ideas achieve performance that is intermediate between the BZ and GV bounds for rates up to about 0.3. We will refer to the resulting rate-distance tradeoff as the Thommesen-Rudra (TR) bound. In [17], the authors implicitly show that that one can achieve *near linear* time decoders up to the TR bound.

In the high rate regime there were no codes, even probabilistic constructions, that were efficiently decodable beyond the BZ bound.

► **Fact 6** (Blokh-Zyablov bound [2, 14]). *For any  $\rho \in (0, \frac{1}{2})$  and any*

$$0 < R < R_{\text{BZ}}(\rho) \stackrel{\text{def}}{=} 1 - H(\rho) - \rho \cdot \int_0^{1-H(\rho)} \frac{dx}{H^{-1}(1-x)},$$

*there exists a rate- $R$  ensemble of codes  $\{C_n\}_{n \in \mathbb{Z}^+}$  that is efficiently  $\rho$ -list decodable and efficiently uniquely-decodable against up to a  $\frac{\rho}{2}$  fraction of errors.*

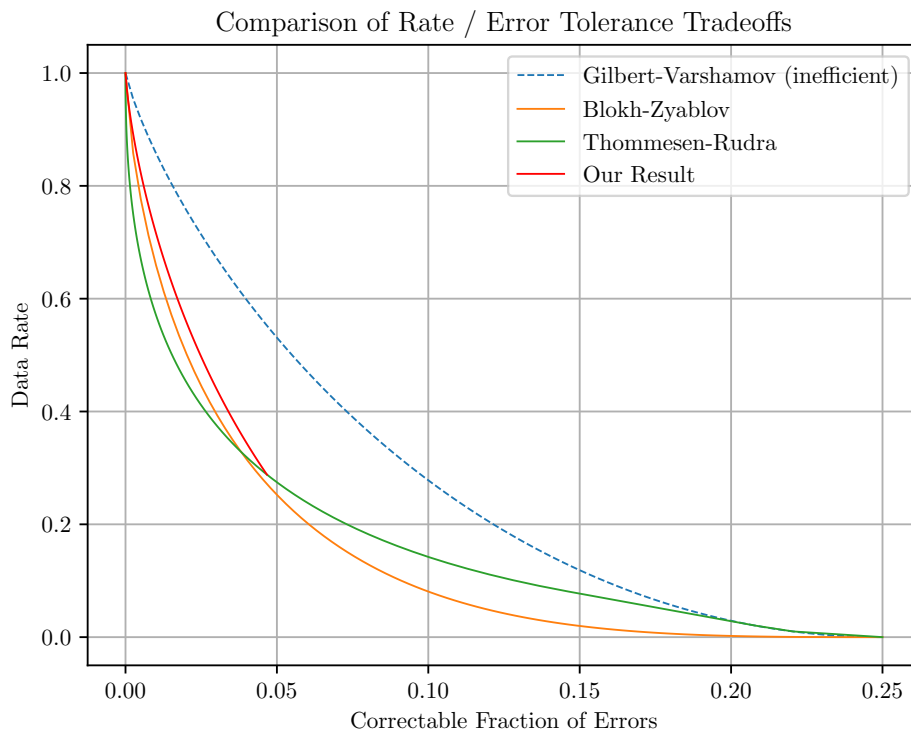
Note that efficient  $\rho$ -list decodability generically implies combinatorial  $(\rho', H(\rho') - H(\rho) + o(1))$ -list decodability for any  $\frac{1}{2} \geq \rho' \geq \rho$ . By combining this with Corollary 4, we obtain bounds that improve over the BZ (and TR) bound for rates above roughly 0.3. This is illustrated in Figure 1.

## 1.4 Related Work

Our work is an application of list decoding, a notion that was introduced by Elias in the 50's [6]. The notion was then (implicitly) revisited with a focus on algorithmic efficiency by Goldreich and Levin [10], who showed how to efficiently list-decode the Hadamard code, and later by Sudan [24], who showed the same for Reed-Solomon codes. List decoding has proven to be a useful notion in computational complexity theory, and has recently been the focus of extensive research (see e.g. the surveys of Sudan [25] and Guruswami [13]).

Several works have studied variants of the error correction problem in which it is possible to obtain improved results on worst-case unique decoding. Guruswami [12] considered a model in which a sender is able send a small amount of information over a noise-free channel, and showed that this enables unique decoding up to the list-decoding radius. Langberg [20] considered the different notion of “private codes”, in which the sender and receiver share some secret randomness, and showed that it is possible to achieve better parameters in this model. Our constructions in contrast use only public randomness, and does not require any noise-free channel.





■ **Figure 1** We improve over previous efficiently decodable binary codes (even probabilistic constructions) [2, 22] for rates above about 0.3. Although it appears in this plot as if the TR bound slightly beats the GV bound for very low rates, this is an artifact of our plotting software that disappears upon zooming in.

Perhaps a more relevant line of work to us is one that studies, loosely speaking, whether imperfectly shared context can improve the efficiency of interactive protocols. This question has been articulated and studied in the settings of interactive communication complexity [3, 7, 8], simultaneous message passing [1], and message compression [18, 16], and in the general setting of “goal-oriented communication” [9].

Our work can be viewed through a similar lens. We seek to improve the efficiency of communication, leveraging context (which implies that only a small number of messages “make sense”). Like in prior works, the context is not fully known to both parties. In fact, we go further: the sender may know *nothing* about the context, other than that the message he is sending makes sense. At the same time, the receiver may know very little about the context – only enough to answer a polynomial number of questions on whether a given message makes sense. Moreover, in contrast to prior works, we do not assume error-free communication channels, and we emphasize the importance of efficient algorithms, while prior works have focused primarily on minimizing communication.

A main idea in this paper is to use list decodable codes, and to permute the message space in such a way as to achieve unique decoding instead of just list decoding. Similar ideas have been used for example in [15] and [4]. However, in those works the adversary is not fully general like in this work, but is restricted (either computationally, or by having to corrupt the codeword in an online fashion).

## 2 Preliminaries

### 2.1 Codes

A deterministic code of dimension  $k$  and block length  $n$  over an alphabet  $\Sigma$  is a (multi-)subset  $\mathcal{C} \subseteq \Sigma^n$  of size  $|\Sigma|^k$ , whose elements are called codewords. The rate of such a code is the quantity  $\frac{k}{n}$ . Throughout this paper, we focus on the case when  $\Sigma$  is a finite field  $\mathbb{F}_q$  and when the dimension  $k$  is integral. In such cases, we associate the codewords of  $\mathcal{C}$  with  $\Sigma^k$ , and we abuse notation by writing  $\mathcal{C}$  to refer both to the multiset of codewords and the corresponding mapping from  $\Sigma^k$  to  $\Sigma^n$ . A code  $\mathcal{C}$  as above is said to be linear if it is a subspace of  $\Sigma^n$ , and in this case the associated mapping can be taken to be a linear function.

For any alphabet  $\Sigma$ , any  $n$ , and any  $u, v \in \Sigma^n$ , the Hamming distance between  $u$  and  $v$ , denoted  $\Delta(u, v)$ , is

$$\Delta(u, v) \stackrel{\text{def}}{=} \left| \{i \in [n] : u_i \neq v_i\} \right|.$$

When  $\Delta(u, v) \leq \delta n$ , we write  $u \approx_\delta v$ . If  $S$  is a set, we write  $\Delta(u, S)$  to denote  $\min_{v \in S} \Delta(u, v)$ . The distance of a code  $\mathcal{C}$  is  $\min_{c \neq c' \in \mathcal{C}} \Delta(c, c')$ .

We also consider probabilistic codes, focusing on codes over binary alphabets.

► **Definition 7.** A probabilistic binary code of block length  $n$  and dimension  $k$  is a randomized function  $\mathcal{C} : \{0, 1\}^k \xrightarrow{\$} \{0, 1\}^n$ .

When discussing the asymptotic performance of (deterministic or probabilistic) codes, it makes sense to consider ensembles of codes  $\{\mathcal{C}_n : \{0, 1\}^{k_n} \rightarrow \{0, 1\}^{\ell_n}\}$  with varying message lengths and block lengths. We will always assume several restrictions on  $k_n$  and  $\ell_n$  to rule out pathological examples. Specifically, we will assume that:

- The limit  $r = \lim_{n \rightarrow \infty} \frac{k_n}{\ell_n}$  exists with  $r \in (0, 1)$ . We call  $r$  the rate of the ensemble.
- $\lim_{n \rightarrow \infty} \frac{\ell_n}{n} = 1$ . This is important so that for a large message of length  $k$ , the cost of padding to length  $k_n$  is not too large.

Given these two assumptions, it is possible without loss of generality to assume  $\ell_n = n$  (we can always take a code from the ensemble with larger  $\ell_n$ , and truncate it; asymptotically, this affects neither its rate nor its error tolerance).

► **Definition 8.** We say that an ensemble of codes  $\{C_n : \{0, 1\}^{k_n} \rightarrow \{0, 1\}^n\}_{n \in \mathbb{Z}^+}$  is combinatorially  $(\rho, \lambda)$ -list decodable if there is some  $L(n) \leq 2^{(\lambda + o(1)) \cdot n}$  and  $\rho'(n) \geq \rho - o(1)$  such that for any  $y \in \{0, 1\}^n$ , there are at most  $L(n)$  values of  $m \in \{0, 1\}^{k_n}$  for which  $C_n(m) \approx_{\rho'(n)} y$ . If there is a polynomial-time algorithm that outputs all such  $m$  (in which case we can assume  $\lambda = 0$ ), then we say that  $\{C_n\}$  is efficiently  $\rho$ -list decodable.

We will also say that  $\{C_n\}$  is combinatorially  $\rho$ -list decodable if it is combinatorially  $(\rho, 0)$ -list decodable.

### 2.2 Binomial Coefficients

We will use the following approximations of binomial coefficients.

► **Fact 9.** For any  $n, k \in \mathbb{Z}^{\geq 0}$ , it holds that

$$\left(\frac{n}{k}\right)^k \leq \binom{n}{k} \leq \left(\frac{en}{k}\right)^k.$$

## 43:10 Error Correcting Codes for Uncompressed Messages

For all constant  $0 \leq \delta \leq 1$ , as  $n$  goes to infinity

$$\binom{n}{\delta n} = \tilde{\Theta}(2^{H(\delta)n}),$$

where  $H(p) = -p \log_2 p - (1-p) \log_2 (1-p)$  is the binary entropy function.

We will also use the standard notion of  $q$ -ary entropy.

► **Definition 10.** The  $q$ -ary entropy function is

$$H_q(x) = x \log_q(q-1) - x \log_q x - (1-x) \log_q(1-x).$$

We define the inverse function  $H_q^{-1}$  to map any  $y \in [0, 1]$  to the unique value  $x \in [0, 1 - 1/q]$  for which  $H_q(x) = y$ .

### 2.3 Covering Numbers for Hamming Balls

For  $x \in \{0, 1\}^n$ , we will denote by  $B_r(x)$  the Hamming ball of radius  $r$  centered at  $x$ , i.e. the set  $\{x' \in \{0, 1\}^n : \Delta(x, x') \leq r\}$ .

► **Definition 11.** Let  $S$  be a subset of  $\{0, 1\}^n$ , and let  $r$  be a positive real number. An  $r$ -covering of  $S$  is a subset  $C$  of  $\{0, 1\}^n$  such that  $S \subseteq \cup_{x \in C} B_r(x)$ . The  $r$ -covering number of  $S$ , denoted  $N_r(S)$ , is the minimum cardinality of any  $r$ -covering of  $S$ .

A volume argument with Fact 9 shows, for any  $0 < \delta_0 < \delta_1 \leq \frac{1}{2}$ , that  $N_{\delta_0 n}(B_{\delta_1 n}) \geq \tilde{\Omega}(2^{(H(\delta_1) - H(\delta_0)) \cdot n})$ . In fact, a simple application of the probabilistic method (due to Dumer et al.) also shows that  $N_{\delta_0 n}(B_{\delta_1 n}) \leq \tilde{O}(2^{(H(\delta_1) - H(\delta_0)) \cdot n})$ . These two statements are combined in the following fact.

► **Fact 12** ([5, Eq. 2.4]). For any  $0 \leq \delta_0 < \delta_1 \leq \frac{1}{2}$  and any  $x \in \{0, 1\}^n$ , it holds that

$$N_{\delta_0 n}(B_{\delta_1 n}(x)) = \tilde{\Theta}(2^{(H(\delta_1) - H(\delta_0)) \cdot n}).$$

### 2.4 $t$ -wise Independence

► **Definition 13.** A family of hash functions  $\{h_i : X \rightarrow Y\}_{i \in \mathcal{I}}$  is said to be  $t$ -wise independent if for all distinct  $x_1, \dots, x_t \in X$ , the distribution of  $(h_i(x_1), \dots, h_i(x_t))$  for a uniformly random  $i \in \mathcal{I}$  is uniformly random over  $Y^t$ .

► **Imported Theorem 14** ([27]). For any  $n, m, t \in \mathbb{Z}^+$ , there exists a  $t$ -wise independent family of hash functions mapping  $\{0, 1\}^n$  to  $\{0, 1\}^m$  such that it takes  $\text{poly}(n, m, t)$  time to sample or evaluate a hash function.

► **Definition 15.** A family of permutations  $\{\pi_i : X \rightarrow X\}_{i \in \mathcal{I}}$  is said to be  $t$ -wise  $\epsilon$ -dependent if for all distinct  $x_1, \dots, x_t \in X$  it holds for uniformly random  $i \in \mathcal{I}$  that the distribution of  $(\pi_i(x_1), \dots, \pi_i(x_t))$  is  $\epsilon$ -close in statistical distance to uniformly random over tuples of distinct  $y_1, \dots, y_t \in X$ .

► **Imported Theorem 16** ([19, Theorem 5.9]). For any  $\epsilon > 0$  and any  $t \in \mathbb{Z}^+$ , there exists a  $t$ -wise  $\epsilon$ -dependent family of permutations on  $\{0, 1\}^n$  with description length  $O(nt + \log(\frac{1}{\epsilon}))$  such that it takes time  $\text{poly}(n, t, \log(\frac{1}{\epsilon}))$  to sample, evaluate, or invert a permutation.

### 3 Contextually Unique Decoding

In this section we present the notion of contextually-unique decoding, and we give some simple constructions of contextually-unique decoders with qualitatively worse parameters than our main result.

► **Definition 17.** An oracle algorithm  $D$  is an  $(r, \delta, \epsilon, \tau)$ -contextually unique decoder for a family of probabilistic codes  $\{C_i : \{0, 1\}^k \xrightarrow{\$} \{0, 1\}^n\}_{i \in \mathcal{I}}$  if for all sets  $S \subseteq \{0, 1\}^k$  with  $|S| \leq 2^{rn}$ , it holds with probability at least  $1 - \epsilon$  over the choice of  $i \leftarrow \mathcal{I}$  that for all messages  $m \in S$  and all  $\delta$ -Hamming adversaries  $\mathcal{A}$ ,

$$\Pr_{c \leftarrow C_i(m)} [D^S(i, \mathcal{A}(c)) \neq m] \leq \tau.$$

#### On the order of quantifiers

In the definitions above, we fix a family of codes, then say that for all small enough  $S$ , a random code from the family is good for  $S$ . In particular, we do not allow an adversary to choose  $S$  after the code is sampled. This may be problematic in some cases, but Definition 17 suffices in the common case of languages that are already established (but not perfectly compressible). This includes languages like “English sentences” or “images of dogs”. In this case, since the set  $S$  is already in principle determined (albeit not fully understood), it suffices to pick and agree upon a random code from the family ahead of time, and always use that code in the future.

#### 3.1 Inefficient Decoding

It is possible to show that for fixed deterministic codes, contextually unique decoding is no easier than unique decoding for the entire ambient message space. In Theorems 18 and 20, we show that randomly sampled codes can do better (albeit with an inefficient decoder).

A family of codes  $\{C_i : \{0, 1\}^k \rightarrow \{0, 1\}^n\}_i$  is said to be **pairwise independent** if for all distinct  $x, x' \in \{0, 1\}^k$ , the distribution of  $(C_i(x), C_i(x'))$  for random  $i$  is uniform over  $\{0, 1\}^n \times \{0, 1\}^n$ . For instance, a random linear code is pairwise independent.

► **Theorem 18.** Let  $\{\mathcal{C}_n\}_{n \in \mathbb{Z}^+}$  be an ensemble of pairwise independent code families<sup>2</sup>, where each code in the family  $\mathcal{C}_n$  has  $n$ -bit codewords. Then for all  $r, \delta \in (0, 1)$  with  $H(2\delta) + 2r < 1$ , there is a  $(r, \delta, \exp(-\Omega(n)), 0)$ -contextually unique decoder for  $\mathcal{C}_n$ .

**Proof.** Let  $S$  be a message space with  $|S| \leq 2^{rn}$ . We will show that with all but  $\exp(-\Omega(n))$  probability over the choice of code  $C \leftarrow \mathcal{C}_n$ , the restriction  $C|_S$  of  $C$  to  $S$  has relative distance  $2\delta$ .

For any distinct  $m, m' \in S$ , it follows from pairwise independence and Fact 9 that

$$\Pr_C[C(m) \approx_{2\delta} C(m')] \leq \frac{\tilde{O}(2^{H(2\delta)n})}{2^n} \leq \frac{1}{\tilde{\Omega}(2^{(1-H(2\delta)) \cdot n})}.$$

Union bounding over all pairs of  $m, m'$ ,

$$\Pr_C[\exists m, m' \in S \text{ s.t. } m \neq m' \text{ and } C(m) \approx_{2\delta} C(m')] \leq \frac{1}{\tilde{\Omega}(2^{(1-2r-H(2\delta)) \cdot n})} \leq \exp(-\Omega(n)). \blacktriangleleft$$

<sup>2</sup> Note that we do not explicitly make any assumption on the rate vs. distance tradeoff of  $\mathcal{C}_n$ ; instead, we implicitly use the fact that any code drawn from a pairwise independent family has relatively good distance with high probability.

### Discussion

One interesting aspect of Theorem 18 is that it demonstrates a family of codes with an “apparent rate” that is independent of the number of tolerable errors, as long as the “true” message space is sufficiently sparse. For example, each  $C_n$  might map  $\{0, 1\}^{2n} \rightarrow \{0, 1\}^n$ , yet as long as the  $2n$ -bit messages have some structure that is known to the receiver (not necessarily to the sender!), it can be guaranteed that the receiver will reconstruct the sender’s message.

However, the parameters achieved by Theorem 18 are not optimal. In particular, its error-tolerance is not competitive with the alternative approach of first compressing messages in  $S$  into  $rn$ -bit representations, and then applying a good error-correcting code to this representation. The GV bound for binary codes states that there exist codes with rate  $r$  and relative distance  $2\delta$  whenever  $H(2\delta) + r < 1$ . It is consistent with current knowledge that this bound is tight.

Our next result closes this gap by sampling a probabilistic code rather than a deterministic code.

► **Construction 19.** Let  $C_1, \dots, C_N : \{0, 1\}^k \rightarrow \{0, 1\}^n$  be deterministic binary codes. We define the probabilistic code  $C_{\text{mix}}[C_1, \dots, C_N] : \{0, 1\}^k \xrightarrow{\$} \{0, 1\}^n$  so that  $C_{\text{mix}}[C_1, \dots, C_N](m)$  is  $C_i(m)$  for a uniformly random  $i \leftarrow [N]$ .

► **Theorem 20.** Let  $\{C_n\}_{n \in \mathbb{Z}^+}$  be an ensemble, where  $C_n$  is a pairwise independent family of codes with  $n$ -bit codewords.

For all  $r, \delta \in (0, 1)$  with  $H(2\delta) + r < 1$  and any  $\tau \geq n^{-O(1)}$ , there exists  $N \leq n^{O(1)}$  such that there is an (inefficient)  $(r, \delta, \exp(-\Omega(n)), \tau)$ -contextually unique decoder for  $\{C_{\text{mix}}[C_1, \dots, C_N]\}_{C_i \in C_n}$ .

In other words,  $C_n$  pairwise independent family of codes with  $n$ -bit codewords, and the code we use is  $\{C_{\text{mix}}[C_1, \dots, C_N]\}_{C_i \in C_n}$ , where the  $C_1, \dots, C_N$  are randomly chosen elements of  $C_n$ .

**Proof Overview.** Suppose that a sender encodes a message  $m$ , and the receiver gets an adversarially perturbed codeword  $c'$ . We define the (inefficient) decoder so that it finds all  $i'$  and all  $m' \in S$  for which  $C_{i'}(m')$  is within distance  $\delta n$  of  $c'$ . We claim that with high probability, the only such  $(i', m')$  is in fact  $(i, m)$ .

To see this, we first fix  $m$ , and consider two different ways in which an encoding of  $m$  can be confused for an encoding of a different message. Using Fact 9 one can show that, for each  $i$ :

1. The probability over the choice of  $C_i$  that there exists  $m' \in S \setminus \{m\}$  such that  $C_i(m')$  and  $C_i(m)$  are within Hamming distance  $2\delta n$  is at most  $2^{(r+H(2\delta)-1)n}$ .
  2. The probability over  $C_1, \dots, C_{i-1}, C_{i+1}, \dots, C_N$  that there exists  $m' \in S \setminus \{m\}$  and  $i' \neq i$  such that  $C_{i'}(m')$  and  $C_i(m)$  are within Hamming distance  $\delta n$  is at most  $N \cdot 2^{(r+H(2\delta)-1)n}$ .
- At this point, unless  $H(2\delta) + 2r < 1$ , we cannot simply apply a union bound to argue that with high probability  $C_i(m)$  and  $C_{i'}(m')$  are  $2\delta n$ -far for all  $m \neq m'$ .

To rely only on the weaker condition that  $H(2\delta) + r < 1$ , the key insight is that for any fixed  $m$ , “most” (all but a  $\tau$  fraction) of  $C_i$ ’s will be good in the above sense with all but  $2^{-(r+\Omega(1)) \cdot n}$  probability. To show this, we must set  $N$  to be a sufficiently large polynomial and use Azuma’s inequality (rather than Chernoff) because the events  $\{(2) \text{ holds for } i\}_i$  are not mutually independent. After this, the probability  $2^{-(r+\Omega(1)) \cdot n}$  is sufficiently small that we can union bound over all  $2^{rn}$  choices of  $m$ . ◀

Rather than elaborating on the details here, we instead defer to our full proof of Theorem 21, which uses the same approach.

## Discussion

Unlike Theorem 18, Theorem 20 matches (other than the arbitrarily small inverse polynomial probability of decoding error) the rate vs. error tolerance tradeoff that is known to be achievable with inefficient decoding for *known, efficiently compressible* sets  $S$  (the GV bound).

## 3.2 Efficient Decoding with Noticeable Error

To obtain an efficient contextually-unique decoder, we adapt the ideas of Theorem 20. Instead of using a pairwise independent family of codes (which is not efficiently decodable), we use a “random” efficiently list-decodable code. Specifically, we use a fixed efficiently list-decodable deterministic code, composed with a random (efficiently evaluable and invertible) permutation  $\pi$ .

Recall the definition of  $C_{\text{mix}}$  from Construction 19.

► **Theorem 21.** *Let  $\{C'_n : \{0,1\}^{k_n} \rightarrow \{0,1\}^n\}_{n \in \mathbb{Z}^+}$  be a rate- $r'$  ensemble of (deterministic) binary codes that is efficiently  $\rho$ -list decodable and combinatorially  $(2\delta, \lambda)$ -list decodable.*

*Then, for any  $r < r' - \lambda$  and any  $\tau(n) \geq n^{-O(1)}$ , there exists  $N(n) \leq n^{O(1)}$  such that for any pairwise independent family  $\Pi_n$  of permutations of  $\{0,1\}^{k_n}$ , the family of codes  $\{C_{\text{mix}}[C'_n \circ \pi_1, \dots, C'_n \circ \pi_N]\}_{\pi_1, \dots, \pi_N \in \Pi_n}$  has a  $(r, \delta, \exp(-\Omega(n)), \tau)$ -contextually unique decoder.*

## Discussion

The main advantage of Theorem 21 over Theorem 20 is that the decoder can run in  $\text{poly}(n)$  time. The main disadvantage compared to Theorem 18 is that the probability of incorrectly decoding is relatively high; in particular, the length of the description of a code (and therefore also the encoder’s and decoder’s running times) are inversely proportional to the error probability.

Our proof of Theorem 21 relies on the following version of the Azuma-Hoeffding inequality, which can be found as Equation (3) in [23]:

► **Imported Theorem 22** (Azuma-Hoeffding). *Let  $\{X_k\}_{k=0}^\infty$  be a real-valued martingale with  $a_k \leq X_k - X_{k-1} \leq b_k$ . Then for every  $r \geq 0$ ,*

$$\Pr[|X_n - X_0| \geq t] \leq 2 \cdot \exp\left(-\frac{2t^2}{\sum_{k=1}^n (b_k - a_k)^2}\right).$$

We now commence the proof of Theorem 21.

**Proof.** We first describe the decoding algorithm. We are given as input a corrupted codeword  $y \in \{0,1\}^n$ , and given oracle access to a set  $S$  of “valid messages”. We run the list-decoding algorithm for  $C'_n$  on  $y$  to obtain a list of codewords  $c_i = C'_n(m_i)$  for  $i = 1, \dots, L$ . We find  $i \in [L], j \in [N]$  that  $\pi_j^{-1}(m_i)$  is in  $S$ . If no such  $(i, j)$  exists, or if multiple such  $(i, j)$  exists, we reject (output  $\perp$ ). Otherwise, we output  $\pi_j^{-1}(m_i)$ .

Let  $p_0$  denote the quantity  $2^{r_n} \cdot \max_{c \in \{0,1\}^n} \Pr_{m \leftarrow \{0,1\}^{r'_n}}[\Delta(C_n(m), c) \leq 2\delta n]$ . Using a union bound, we can see that  $p_0$  bounds the probability, for any fixed  $c$ , that  $C'_n(y)$  is  $2\delta n$ -close to  $c$  for any of  $2^{r_n}$  different uniformly random  $y$ . The combinatorial list-decodability

## 43:14 Error Correcting Codes for Uncompressed Messages

of  $\{C'_n\}$  implies that  $p_0 \leq 1/\tilde{\Omega}(2^{(r'-r-\lambda) \cdot n})$ . By assumption on  $r$ , this decreases exponentially with  $n$ , and in particular for any  $N \leq n^{O(1)}$ , it holds that

$$\tau \geq \omega(p_0 \cdot N^2). \quad (1)$$

Let  $N(n)$  be a sufficiently large polynomial such that  $2\tau^2 N - \ln(2) \cdot rn \geq \Omega(n)$ .

▷ **Claim 23.** For any permutations  $\pi_1, \dots, \pi_N$ , let  $C_{\pi_1, \pi_2, \dots, \pi_N}$  denote  $C_{\text{mix}}[C'_n \circ \pi_1, \dots, C'_n \circ \pi_N]$ . For every  $m \in S$ , it holds that

$$\begin{aligned} \Pr_{\substack{\pi_1, \dots, \pi_N \\ C := C_{\pi_1, \pi_2, \dots, \pi_N}}} \left[ \begin{array}{l} \exists \delta\text{-Hamming adversary } \mathcal{A} \text{ s.t.} \\ \Pr_{c \leftarrow C(m)}[D^S(\mathcal{A}(c)) \neq m] \geq 3\tau \end{array} \right] &\leq e^{-(2-o(1))\tau^2 N} \\ &\leq 2^{-(r+\Omega(1))n}. \end{aligned}$$

**Proof.** Consider the probability space defined by sampling  $\pi_1, \dots, \pi_N \leftarrow \Pi$ . For each  $i \in [N]$ , define random variables

$$X_i^{(<)} = \begin{cases} 1 & \text{if } \exists j < i \text{ and } \exists m' \in S \text{ s.t. } C_n(\pi_j(m')) \approx_{2\delta} C_n(\pi_i(m)) \\ 0 & \text{otherwise.} \end{cases}$$

Define random variables  $\{X_i^{(=)}\}_{i \in [N]}$  and  $\{X_i^{(>)}\}_{i \in [N]}$  analogously – that is, replace the condition “ $j < i$ ” by “ $j = i$ ” or “ $j > i$ ” respectively.

Note that  $X_1^{(=)}, \dots, X_N^{(=)}$  are mutually independent because  $X_i^{(=)}$  depends only on  $\pi_i$ . The pairwise independence of  $\Pi$  and a union bound over all  $m'$  implies that for each  $i$ ,  $\Pr[X_i^{(=)} = 1] \leq p_0 \leq N \cdot p_0$ .

The random variables  $X_1^{(<)}, \dots, X_N^{(<)}$  are not independent. However, conditioned on  $X_1^{(<)}, \dots, X_{i-1}^{(<)}$  (indeed on any value of  $\pi_1, \dots, \pi_{i-1}$ ) the pairwise independence of  $\pi_i$  and a union bound over  $j < i$  and over  $m'$  implies that

$$\Pr[X_i^{(<)} = 1 | X_1^{(<)}, \dots, X_{i-1}^{(<)}] \leq i \cdot p_0 \leq N \cdot p_0.$$

Similarly,

$$\Pr[X_i^{(>)} = 1 | X_{i+1}^{(>)}, \dots, X_N^{(>)}] \leq (N - i) \cdot p_0 \leq N \cdot p_0.$$

Azuma's inequality (Imported Theorem 22) implies that

$$\begin{aligned} \Pr\left[\sum_i X_i^{(<)} \geq \tau N\right] &\leq 2e^{-2(\tau - p_0 N^2)^2 N} \\ &\leq e^{-(2-o(1))\tau^2 N} \quad \text{by (1),} \end{aligned}$$

and we obtain the same bound on  $\Pr[\sum_i X_i^{(>)} \geq \tau N]$  and  $\Pr[\sum_i X_i^{(=)} \geq \tau N]$ . So

$$\Pr\left[\sum_i (X_i^{(<)} + X_i^{(=)} + X_i^{(>)}) \geq 3\tau N\right] \leq 3 \cdot e^{-(2-o(1))\tau^2 N} \leq e^{-(2-o(1))\tau^2 N},$$

which is equivalent to the statement of the claim. ◁

Theorem 21 follows from union bounding over all  $2^{rn}$  values of  $m \in S$ . ◀



#### 4 Main Theorem: Efficient Decoding with Negligible Error

In our previous constructions, we always had some inverse polynomial probability (over the choice of encoding randomness) of incorrectly decoding. We now show how to reduce this error probability to negligible by using a super-polynomial number of permutations, but preserving the polynomial-time efficiency of encoding and decoding. This is Theorem 25 below, from which Theorem 3 follows immediately (after using Theorem 14 and Theorem 16).

► **Construction 24.** Let  $C' : \{0, 1\}^{r'n} \rightarrow \{0, 1\}^n$  be a deterministic code, let  $\Pi = \{\pi_k : \{0, 1\}^{r'n-s} \rightarrow \{0, 1\}^{r'n-s}\}_{k \in \mathcal{K}}$  be a family of efficiently evaluable and invertible permutations, and let  $h : \{0, 1\}^s \rightarrow \mathcal{K}$  be a hash function.

We define a probabilistic code  $C_{C', \Pi, h} : \{0, 1\}^{r'n-s} \xrightarrow{\$} \{0, 1\}^n$  that encodes a message  $m \in \{0, 1\}^{r'n-s}$  by picking  $x \leftarrow \{0, 1\}^s$  at random, and outputting  $C'(\pi_{h(x)}(m), x)$ .

► **Theorem 25.** Suppose that:

- $C' : \{0, 1\}^{r'n} \rightarrow \{0, 1\}^n$  is a (deterministic) binary code that is efficiently  $\rho_e$ -list-decodable
- $\delta \leq \rho_e$  and  $\lambda$  are such that  $C'$  is combinatorially  $(2\delta, \lambda)$ -list decodable.
- For some  $t = t(n)$  and  $s = s(n)$  satisfying  $\Omega(n) \leq t(n) \leq n^{O(1)}$  and  $\omega(\log n) \leq s(n) \leq o(n)$ :
  - $\Pi = \{\pi_k : \{0, 1\}^{r'n-s} \rightarrow \{0, 1\}^{r'n-s}\}_{k \in \mathcal{K}}$  is a  $(t+1)$ -wise  $\epsilon$ -dependent family of permutations with  $\epsilon \leq 2^{-n}$ , and
  - $\mathcal{H} = \{h_i : \{0, 1\}^s \rightarrow \mathcal{K}\}_{i \in \mathcal{I}}$  is a  $2t$ -wise independent hash family.

Then the family  $\{C_{C', \Pi, h}\}_{h \in \mathcal{H}}$  has an  $(r, \delta, \exp(-\omega(n)), \frac{t}{2^s})$ -contextually unique decoder for any  $r < r' - \lambda$ .

**Proof.** Let  $S \subseteq \{0, 1\}^{r'n-s}$  be any set of messages with  $|S| \leq 2^{rn}$ . We describe the contextually unique decoding algorithm on input  $y \in \{0, 1\}^n$ . First, the algorithm applies the efficient list-decoding algorithm to obtain all codewords  $y'_1, \dots, y'_L$  of  $C'$  that are within relative Hamming distance  $\rho_e$  of  $y$ . Then each  $y'_i$  is parsed as  $(\pi_{h(x)}(m_i), x)$ . The decoding algorithm outputs any  $m_i$  that is in  $S$ . It is immediate from efficient list-decodability that there is at least one such  $m_i$ . We need to show that with high probability there is at most one such  $m_i$ .

We will rely on the following variant of the Chernoff bound for binary random variables, which does not require the random variables to be fully independent. Instead, it only requires bounding the probability that relatively small subsets of variables are simultaneously 1.

► **Imported Theorem 26** ([21]). Let  $X_1, \dots, X_N$  be  $\{0, 1\}$ -valued random variables, let  $0 < \beta < 1$ , and let  $0 < t < \beta N$ . Then

$$\Pr \left[ \sum_{i=1}^N X_i \geq \beta N \right] \leq \frac{1}{\binom{\beta N}{t}} \cdot \sum_{A \in \binom{[N]}{t}} \mathbb{E} \left[ \prod_{i \in A} X_i \right].$$

We will write  $N$  to denote  $2^s$ , and for brevity of notation we will view any hash function  $h \in \mathcal{H}$  directly as the corresponding tuple of permutations  $(\pi_{k_0}, \dots, \pi_{k_{N-1}})$ , where  $k_i = h(i)$ .

It is sufficient to show that for  $\tau(n) = \frac{t}{N}$ , it holds for every  $m \in S$  that

$$\Pr_{(\pi_1, \dots, \pi_N) \leftarrow \mathcal{H}} \left[ \left| \{i : \exists j \in [N], m' \in S \setminus \{m\} \text{ s.t. } C'(\pi_i(m), i) \approx_{2\delta} C'(\pi_j(m'), j)\} \right| \geq \tau \cdot N \right]$$

is at most  $2^{-rn} \cdot \exp(-\omega(n))$ .

## 43:16 Error Correcting Codes for Uncompressed Messages

We will use the Chernoff variant to prove the above inequality. Let  $X_i$  denote the indicator random variable for the event

$$\exists j \in [N], m' \in S \setminus \{m\} \text{ s.t. } C'(\pi_i(m), i) \approx_{2\delta} C'(\pi_j(m'), j),$$

so what we want to bound is  $\Pr \left[ \sum_{i=1}^N X_i \geq \tau \cdot N \right]$ .

For  $i, j \in [N]$  and  $m' \in S \setminus \{m\}$ , let  $Y_{i,j,m'}$  denote the indicator random variable for the event

$$C'(\pi_i(m), i) \approx_{2\delta} C'(\pi_j(m'), j).$$

Let  $A \subseteq [N]$  be a subset of size  $|A| = t$ . Say  $A = \{a_1, \dots, a_t\}$ . We have

$$\begin{aligned} \mathbb{E} \left[ \prod_{a \in A} X_a \right] &\leq \mathbb{E} \left[ \prod_{a \in A} \sum_{\substack{j \in [N] \\ m' \in S \setminus \{m\}}} Y_{a,j,m'} \right] \\ &= \sum_{\substack{j_1, \dots, j_t \in [N] \\ m'_1, \dots, m'_t \in S \setminus \{m\}}} \mathbb{E} \left[ \prod_{i=1}^t Y_{a_i, j_i, m'_i} \right]. \end{aligned} \quad (2)$$

We now would like to use the independence of  $\mathcal{H}$  and of  $\Pi$  to equate  $\mathbb{E} \left[ \prod_i Y_{a_i, j_i, m'_i} \right]$  with  $\prod_i \mathbb{E}[Y_{a_i, j_i, m'_i}]$ . However this is not quite true, for two reasons. First,  $\Pi$  is only *approximately*  $(t+1)$ -wise independent. Second,  $\Pi$  is a family of  $(t+1)$ -wise (almost) independent *permutations*, rather than unstructured functions.

Still, an only slightly worse bound holds for  $\mathbb{E} \left[ \prod_{i=1}^t Y_{a_i, j_i, m'_i} \right]$ . Conditioned on  $\pi_{j_1}(m'_1), \dots, \pi_{j_t}(m'_t)$  and  $\pi_{a_1}(m), \dots, \pi_{a_{t-1}}(m)$ , the  $2t$ -wise independence of  $\mathcal{H}$  and the  $(t+1)$ -wise  $\epsilon$ -dependence of  $\Pi$  imply that the distribution of  $\pi_{a_t}(m)$  is  $(\epsilon + \frac{t}{2^{r'n-s}})$ -close to uniform over  $\{0, 1\}^{r'n-s}$ . The combinatorial list-decodability of  $C'$  asserts that the number of  $y$  for which  $C'(y) \approx_{2\delta} C'(\pi_{j_i}(m'), j_i)$  is at most  $2^{\lambda \cdot n}$ .

We can therefore continue bounding (2) as follows:

$$\begin{aligned} &\leq \sum_{\substack{j_1, \dots, j_t \in [N] \\ m'_1, \dots, m'_t \in S \setminus \{m\}}} \prod_{i=1}^t \left( \frac{2^{\lambda \cdot n}}{2^{r'n-s}} + \epsilon + \frac{t}{2^{r'n-s}} \right) \\ &\leq (N \cdot 2^{rn})^t \cdot 2^{(\lambda - r' + o(1)) \cdot nt} \\ &\leq \alpha(n)^t, \end{aligned}$$

where we define  $\alpha(n) = N(n) \cdot \tilde{O}(2^{(\lambda + r - r' + o(1)) \cdot n})$ , which is  $\exp(-\Omega(n))$  by assumption on  $\delta$  and  $r$  and because  $N \leq 2^{o(n)}$ . Thus for  $\tau(n) = \frac{t}{N} \geq \omega(\alpha(n))$ , it holds by Imported Theorem 26 that

$$\begin{aligned} \Pr \left[ \sum_{i=1}^N X_i \geq \tau \cdot N \right] &\leq \frac{\binom{N}{t}}{\binom{\tau \cdot N}{t}} \cdot \alpha(n)^t \\ &\leq \left( \frac{e\alpha}{\tau} \right)^t \\ &\leq \exp(-\omega(t)) \\ &\leq \exp(-\omega(n)) \\ &\leq 2^{rn} \cdot \exp(-\omega(n)). \end{aligned}$$

Theorem 25 follows by union bounding over all  $2^{rn}$  choices of  $m \in S$ . ◀

## 5 Future Directions

There are several interesting directions that we have not yet explored. We highlight a few below:

- How well is it possible to perform contextually unique decoding in different error models? For example, one might consider adversarial erasures, insertions, deletions, random errors, and so on.
- What are the optimal achievable parameters for contextually unique decoding?
- Is it possible to have a single probabilistic code that simultaneously works well for all message sets  $S$  of bounded size? If so, with what parameters?
- When  $S = \{0, 1\}^k$  padded with zeroes, can our construction be made explicit?

---

## References

- 1 Mohammad Bavarian, Dmitry Gavinsky, and Tsuyoshi Ito. On the role of shared randomness in simultaneous communication. In *International Colloquium on Automata, Languages, and Programming*, pages 150–162. Springer, 2014.
- 2 E. L Blokh and Victor Zyablov. Linear concatenated codes. *Nauka*, 1982.
- 3 Clément L Canonne, Venkatesan Guruswami, Raghu Meka, and Madhu Sudan. Communication with imperfectly shared randomness. *IEEE Transactions on Information Theory*, 63(10):6799–6818, 2017.
- 4 Zitan Chen, Sidharth Jaggi, and Michael Langberg. The capacity of online (causal)  $q$ -ary error-erasure channels. *IEEE Transactions on Information Theory*, 65(6):3384–3411, 2019.
- 5 Ilya Dumer, Mark S. Pinsker, and Vyacheslav V. Prelov. Epsilon-entropy of an ellipsoid in a hamming space. *Probl. Inf. Transm.*, 38(1):1–15, 2002.
- 6 Peter Elias. List decoding for noisy channels. Technical Report 335, Research Laboratory of Electronics, MIT, 1957.
- 7 Badih Ghazi, Ilan Komargodski, Pravesh Kothari, and Madhu Sudan. Communication with contextual uncertainty. In *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, pages 2072–2085. SIAM, 2016.
- 8 Badih Ghazi and Madhu Sudan. The power of shared randomness in uncertain communication. In *ICALP*, volume 80 of *LIPIcs*, pages 49:1–49:14. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017.
- 9 Oded Goldreich, Brendan Juba, and Madhu Sudan. A theory of goal-oriented communication. *Journal of the ACM (JACM)*, 59(2):8, 2012.
- 10 Oded Goldreich and Leonid A Levin. A hard-core predicate for all one-way functions. In *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 25–32. ACM, 1989.
- 11 Ofer Grossman and Justin Holmgren. Error correcting codes for uncompressed messages. *Electron. Colloquium Comput. Complex.*, 27:38, 2020.
- 12 Venkatesan Guruswami. List decoding with side information. In *IEEE Conference on Computational Complexity*, page 300. IEEE Computer Society, 2003.
- 13 Venkatesan Guruswami. Algorithmic results in list decoding. *Theoretical Computer Science*, 2(2):107–195, 2006.
- 14 Venkatesan Guruswami and Atri Rudra. Better binary list decodable codes via multilevel concatenation. *IEEE Transactions on Information Theory*, 55(1):19–26, 2009.
- 15 Venkatesan Guruswami and Adam Smith. Optimal rate code constructions for computationally simple channels. *Journal of the ACM (JACM)*, 63(4):1–37, 2016.
- 16 Elad Haramaty and Madhu Sudan. Deterministic compression with uncertain priors. *Algorithmica*, 76(3):630–653, 2016.
- 17 Brett Hemenway, Noga Ron-Zewi, and Mary Wootters. Local list recovery of high-rate tensor codes & applications. In *FOCS*, pages 204–215. IEEE Computer Society, 2017.

- 18 Brendan Juba, Adam Tauman Kalai, Sanjeev Khanna, and Madhu Sudan. Compression without a common prior: an information-theoretic justification for ambiguity in language. In *ICS*, pages 79–86. Tsinghua University Press, 2011.
- 19 Eyal Kaplan, Moni Naor, and Omer Reingold. Derandomized constructions of  $k$ -wise (almost) independent permutations. *Algorithmica*, 55(1):113–133, 2009.
- 20 Michael Langberg. Private codes or succinct random codes that are (almost) perfect. In *FOCS*, pages 325–334. IEEE Computer Society, 2004.
- 21 Nathan Linial and Zur Luria. Chernoff’s inequality - a very elementary proof, 2014. [arXiv:1403.7739](#).
- 22 Atri Rudra. *List decoding and property testing of error correcting codes*. PhD thesis, University of Washington, 2007.
- 23 Igal Sason. On refined versions of the Azuma-Hoeffding inequality with applications in information theory. *arXiv preprint*, 2011. [arXiv:1111.1977](#).
- 24 Madhu Sudan. Decoding of Reed Solomon codes beyond the error-correction bound. *Journal of complexity*, 13(1):180–193, 1997.
- 25 Madhu Sudan. List decoding: Algorithms and applications. In *IFIP International Conference on Theoretical Computer Science*, pages 25–41. Springer, 2000.
- 26 C. Thommesen. The existence of binary linear concatenated codes with Reed-Solomon outer codes which asymptotically meet the Gilbert-Varshamov bound. *IEEE Transactions on Information Theory*, 29(6):850–853, November 1983. doi:10.1109/tit.1983.1056765.
- 27 Mark N. Wegman and Larry Carter. New hash functions and their use in authentication and set equality. *J. Comput. Syst. Sci.*, 22(3):265–279, 1981.
- 28 John M Wozencraft. List decoding. *Quarterly Progress Report*, 48:90–95, 1958.

# Total Functions in the Polynomial Hierarchy

**Robert Kleinberg**

Cornell University, Ithaca, NY, USA

**Oliver Korten**

Columbia University, New York, NY, USA

**Daniel Mitropolsky**

Columbia University, New York, NY, USA

**Christos Papadimitriou**

Columbia University, New York, NY, USA

---

## Abstract

We identify several genres of search problems beyond **NP** for which existence of solutions is guaranteed. One class that seems especially rich in such problems is **PEPP** (for “polynomial empty pigeonhole principle”), which includes problems related to existence theorems proved through the union bound, such as finding a bit string that is far from all codewords, finding an explicit rigid matrix, as well as a problem we call **COMPLEXITY**, capturing Complexity Theory’s quest. When the union bound is generous, in that solutions constitute at least a polynomial fraction of the domain, we have a family of seemingly weaker classes  $\alpha$ -**PEPP**, which are inside  $\mathbf{FP}^{\mathbf{NP}}|_{\text{poly}}$ . Higher in the hierarchy, we identify the constructive version of the Sauer-Shelah lemma and the appropriate generalization of **PPP** that contains it, as well as the problem of finding a king in a tournament (a vertex  $k$  such that all other vertices are defeated by  $k$ , or by somebody  $k$  defeated).

**2012 ACM Subject Classification** Theory of computation → Complexity classes

**Keywords and phrases** total complexity, polynomial hierarchy, pigeonhole principle

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.44

**Funding** *Daniel Mitropolsky*: This research was supported in part by a grant from the Columbia-IBM center for Blockchain and Data Transparency, and by JPMorgan Chase & Co. This research was supported in part by NSF Awards CCF1763970 and CCF1910700, and by a research contract with Softbank

*Christos Papadimitriou*: This research was supported in part by NSF Awards CCF1763970 and CCF1910700, and by a research contract with Softbank.

**Acknowledgements** Many thanks to Noga Alon for a very enlightening conversation about the union bound, and to Omri Weinstein for an interesting discussion. The authors also thank Ofer Grossman and Eylon Yogeve for useful discussions after the pre-print.

## 1 Introduction

The complexity of total functions has emerged over the past three decades as an intriguing and productive branch of Complexity Theory. Subclasses of **TFNP**, the set of all total functions in **FNP**, have been defined and studied: **PLS**, **PPP**, **PPA**, **PPAD**, **PPADS**, and **CLS**. These classes are replete with natural problems, several of which turned out to be complete for the corresponding class, see e.g. [9, 10].

Each of these classes corresponds naturally to a very simple existential argument. For example, **PLS** is the class of all total functions whose proof of totality relies on the fact that *every finite dag must have a sink*, while **PPAD** captures this true existential statement: “*If a finite directed graph has an unbalanced node (i.e., a node whose in-degree differs from its out-degree), then it must have another unbalanced node.*” The class of total functions **PPP** (for “polynomial pigeonhole principle”) captures the well known fact that “*if there are more*



© Robert Kleinberg, Oliver Korten, Daniel Mitropolsky, and Christos Papadimitriou;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 44; pp. 44:1–44:18

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

*pigeons than pigeonholes, there must be a pigeonhole with two or more pigeons.*” This latter complexity class has attracted much attention due to its close connections to cryptography, and there has been recent progress towards natural complete problems [5, 22, 14].

More recently a logic-inspired class **PTFNP** (for “provable **TFNP**”) was identified containing all of the above classes [11], its definition motivated by the existential proof point of view described above. It was also pointed out in [11] that finitariness is necessary for the definition of a meaningful class of total functions, in that any non-finitary existence theorem – that is, one that also holds for infinite structures – results in a computational problem that is provably easy. Also recently, an intriguing link between the possibility of **TFNP**-hardness and average-case hardness was discovered [12].

The simple statement on which **PPP** is based has a very natural “dual” variant, call it the *empty pigeonhole principle*, namely: “*if there are more pigeonholes than pigeons, then there must be an empty pigeonhole.*” Concretely, given a circuit  $C$  mapping  $[2^n - 1]$  to  $[2^n]^1$ , find a bit string of length  $n$  that is not in  $C$ ’s range. Call this problem **EMPTY**. One could even define a class based on the empty pigeonhole principle, call it **PEPP** (for “polynomial empty pigeonhole principle,” the set of all total function problems polynomial-time reducible to **EMPTY**). At first sight, **PEPP** may seem very close to **PPP** – identical, perhaps? – *until one notices that **PEPP** is not obviously in NP!* For **PPP**, one can guess and check the offending pigeonhole *and* the two pigeons in it – but for **PEPP**? Once the empty pigeonhole has been guessed, proving it is empty requires one to look at all pigeons. An alternation of quantifiers appears to be at work!

*In this paper we introduce a hierarchy of total search problems analogous to the polynomial hierarchy of decision problems.* **TFNP** is the first level of the hierarchy, and the class **PEPP** just defined is at the second level of this hierarchy, denoted **TFΣ<sub>2</sub>**. Actually, we shall soon see that there are natural and interesting search problems occupying the third level of the hierarchy. (For the formal definition of **TFΣ<sub>i</sub>** and some basic facts about this hierarchy, see the Appendix.)

The first result we prove in this direction is that, despite the apparent similarity and “symmetry” outlined above, **PEPP** contains **PPP** – *and in fact, all of FNP* (Theorem 1; the proof is easy).

**EMPTY** and **PEPP** are closely associated with the familiar probabilistic argument known as *the union bound*. There is a formal way to see this: Consider a generic instance of **EMPTY**, that is, a circuit  $C$  mapping  $[2^n - 1]$  to  $[2^n]$ ; the task is to find a possible output in  $[2^n]$  not in the circuit’s range. Interpret now an input  $x$  as  $x = yz$ , where  $|z| = n - m$  and  $|y| = m$ , and where  $y$  encodes a “bad event” – in the sense of the union bound – with probability  $2^{-m}$  ( $2^{-m} - 2^{-n}$  for one of the events), while  $z$  indexes the  $2^{n-m}$  (respectively,  $2^{n-m} - 1$ ) elements of the whole probability space of size  $2^n$  that constitute the bad event. Hence, the empty pigeonhole principle can be interpreted as the union bound. Many of the important natural problems in **PEPP** correspond to existential proofs through the union bound, or more generally through counting.

One of these problems is **REMOTE POINT**: Given a code – generically, a circuit mapping  $[2^k]$  to  $[2^n]$  where  $k < n$  and the codewords are the range of the circuit – find an  $n$ -bit string that is far from all codewords (as far in Hamming distance, that is, as is guaranteed by the union bound). It is not hard to see that **REMOTE POINT** is **PPEP**-complete. The important open problem here is the complexity of the special case of **REMOTE POINT** in which the circuit is a linear function in **GF<sub>2</sub>**; this is a much studied problem [2].

---

<sup>1</sup> We denote the set  $\{0, 1, \dots, M - 1\}$  by  $[M]$ . We shall see that it is easy to construct circuits with arbitrary integer domains and ranges.

Other natural problems in **PEPP** capture interesting aspects of *complexity*. To start with the more indirect one, RIGID MATRIX COMPLETION is the following problem: find a rigid matrix in  $\mathbf{GF}_2^{n \times n}$  (that is, an  $n \times n$  matrix whose rank cannot collapse to something tiny by manipulating very few entries, details supplied later) given several of its entries. Rigid matrices have been shown [24] to be abundant, and to capture logarithmic-depth circuit complexity, while their explicit construction has remained an important open problem since Valiant's paper. We define the relevant problem as a *completion* variant of the explicit construction problem – that is, part of the matrix is specified – to overcome a familiar impediment: without a binary input of some substantial length, one is dealing with a *sparse* problem, and current techniques seem unable to fathom the complexity of such problems (see the related work subsection for a further discussion of this). A second problem in the same vein, RAMSEY-ERDŐS COMPLETION, embodies Erdős's famous 1959 proof that the  $n$ -th Ramsey number is at least  $2^{\frac{n}{2}}$ .

COMPLEXITY is a problem asking, given a bit string of length  $n$ , to find an explicit Boolean function with  $\log n$  inputs which requires  $\Omega(\frac{n}{\log^2 n})$  gates – that is, an explicit exponential lower bound. The problem is, again, defined through a circuit. The circuit interprets its input gates as the representation of a circuit with  $\log n$  inputs and  $O(\frac{n}{\log^2 n})$  gates, where besides the usual Boolean gates we also allow *oracle gates* with fan-in  $\log n$ . The output of the circuit is the Boolean function computed by this circuit, encoded as a bit string of length  $2^{\log n} = n$ . The input to the problem (on the basis of which the circuit is constructed and the computation of the circuit is carried out) is also interpreted as an *oracle*, encoding in its  $n$  bits the answers to all possible oracle inputs. The task is to discover an  $n$ -bit string that is not in the range of this circuit under this oracle – that is to say, a Boolean function with  $\log n$  variables which therefore requires  $\Omega(\frac{n}{\log^2 n})$  gates to be computed with the given oracle. The oracle here is needed, again, to render a sparse function exponentially dense.

Is this problem **PEPP**-complete, or otherwise hard in a demonstrable sense? This is an important problem left open in this paper. We are aware of one immediate obstacle: it turns out that many of the problems we discussed above, COMPLEXITY among them, belong in a significantly weakened subclass of **PEPP**. Let  $\alpha$  be a positive quantity, possibly a function of  $n$ , and define the class  $\alpha$ -**PEPP** (pronounced *abundant PEPP*) to be the variant in which the given circuit does not map  $[2^n]$  to  $[2^n + 1]$ , but instead  $[2^n]$  to  $[(1 + \alpha) \cdot 2^n + 1]$ ; evidently, **PEPP** = 0-**PEPP**, while many of the problems in **PEPP** discussed are known to belong to  $\alpha$ -**PEPP** for some constant  $\alpha$ . In particular, we denote 1-**PEPP** by **APEPP**, circuits with twice as many outputs than inputs.

We prove two theorems on  $\alpha$ -**PEPP**. First, we establish that the precise value of  $\alpha$  is in some sense irrelevant, in that any class  $\alpha$ -**PEPP** with  $\alpha$  between  $\frac{1}{\text{poly}}$  and  $\text{poly}$  can be reduced to any other such class *through* **FP<sup>NP</sup>** reductions (Theorem 7; it is not known whether polynomial time reductions are possible here). Second, it turns out that for  $\alpha \geq \frac{1}{\text{poly}}$ , for any problem in  $\alpha$ -**PEPP** with  $n$  input gates there is a small set of outputs (strings of length  $\lceil n \log(1 + \alpha) \rceil$ ) such that, for any input, one of them is an empty pigeonhole. (The proof is by – what else? – the union bound.) It follows that  $\alpha$ -**PEPP** is contained in **FZPP<sup>NP</sup>** – *functional ZPP* (Monte Carlo algorithms) with a satisfiability oracle – and, analogously to Adleman's theorem [1], that  $\alpha$ -**PEPP** is contained in **FP<sup>NP</sup>**|poly, **FP<sup>NP</sup>** with polynomial advice; we see no reason why **PEPP** = 0-**PEPP** should be so confined.

So far we have been discussing problems and classes in **TFΣ<sub>2</sub>**, the next level after **TFNP** of what can be called the polynomial total function hierarchy. It turns out that there are also interesting problems further up. SHATTERING is the following problem: we are given a circuit  $C$  with  $k$  input gates and  $n$  output gates, which is supposed to represent a family of



$2^k$  subsets of  $[n]$ . We must return either a collision in this circuit, establishing that the family has fewer than  $2^k$  distinct sets; or otherwise a  $d$ -subset of  $[n]$ , call it  $D$ , which is *shattered* by the family – that is, every subset of  $D$  can be written as  $D \cap C(x)$  for some set  $C(x)$  in the family; such a set is guaranteed to exist by the Sauer-Shelah lemma [19, 21, 25], as long as  $C$  has no collisions and  $k$  is large enough as a function of  $n$  and  $d$ . Notice immediately that there are two alternations of quantifiers in this existential result: *there is* a set  $D$  such that *for every* subset  $G$  of  $D$  *there is* an output  $C(x)$  of  $C$  such that  $G = D \cap C(x)$ : we are in the class  $\mathbf{TF}\Sigma_3$ ! In fact, we show that SHATTERING belongs to a very natural subclass of  $\mathbf{TF}\Sigma_3$ : it belongs to  $\mathbf{PPP}^{\Sigma_2}$ , the pigeonhole principle class when the function mapping pigeons to pigeonholes can use a  $\Sigma_2$  oracle in its computations.

We present another  $\mathbf{TF}\Sigma_3$  problem, which we denote as KING: given a tournament (succinctly described by a circuit), find a vertex  $v$  such that every other vertex is reachable from  $v$  by a directed path of length one or two. The proof that such a vertex must exist is a local-search argument dating back to the 1950's [15], but the potential function used in the proof is  $\#\mathbf{P}$ -hard to compute, hence the KING problem does not evidently belong to any natural subclass of  $\mathbf{TF}\Sigma_3$  such as  $\mathbf{PLS}^{\Sigma_2}$ .

## Related Work

The difficulty of making existential arguments based on the union bound *constructive* has in fact been a fundamental problem in Complexity Theory and combinatorics for over seven decades. Already in 1947, after Erdős published his paper proving Ramsey lower bounds via the probabilistic method, he recognized the difficulty of matching this with a constructive proof, and offered a \$100 prize to anyone who could do so [8]. Two years after that, Shannon used the union bound to give a nonconstructive proof that some functions require exponential size circuits, and also noted the difficulty in finding constructive proofs of size lower bounds for explicit functions, comparing it to the difficulty of proving that particular numbers are transcendental [20]. At this time, “constructive” was a rather informal concept, but a few decades later Complexity Theory offered us a plausible definition: a constructive proof is an algorithm that constructs an object with the desired properties from scratch, in time polynomial in the size of the object. Over time, an important research tradition in Complexity Theory has developed around such explicit construction problems, pertaining mostly to the construction of computational devices (pseudorandom generators, randomness extractors, exponentially hard Boolean functions in the worst or average case, etc.) whose existence is guaranteed by the union bound. Many celebrated results in this domain compare the difficulty of such explicit construction problems through, essentially, reductions [24, 17, 13, 6, 23]. In particular, already in 1977 Les Valiant [24] showed that an explicit construction of a rigid matrix would imply an explicit Boolean function requiring shallow circuits of superlinear size – a reduction between two explicit construction problems whose corresponding existence proofs rely on the union bound. Next, Nisan and Wigderson established in 1995 that an explicit pseudorandom generator can be constructed in polynomial time, given an explicit construction of a truth table which is hard to approximate for exponentially large circuits, and Impagliazzo and Wigderson [13] showed a decade later that such hard-to-approximate truth tables can in fact be constructed in polynomial time from truth tables which are very hard to compute in the worst case. More recently, an equivalence has been shown between the explicit construction of randomness dispersers and the construction of Ramsey graphs, and a significant body of work has been devoted to deriving more efficient constructions of such objects [8, 6].

Many results in this realm can be reformulated as reductions between total function problems in a particular subclass of **APEPP**, which could be called **SAPEPP** (for “sparse **APEPP**”). Every problem in **SAPEPP** is defined by a polynomial-time Turing machine-computable function  $M : \{0, 1\}^* \mapsto \{0, 1\}^*$  such that for all  $x \in \{0, 1\}^*$ ,  $|M(x)| = f(|x|)$ , where  $f(n) > n$ . The total search problem associated with  $M$  asks: given  $n$  in unary, find a bit string  $y$  of length  $f(n)$  such that for all  $x$  with  $|x| = n$ ,  $M(x) \neq y$ . That **SAPEPP** is a subset of **APEPP** follows easily from the basic fact that any fixed polynomial-time Turing machine with a given input length can be rendered as a Boolean circuit in time polynomial in the input length. For a concrete example, for the problem of explicitly constructing a truth table for a function  $[N] \rightarrow \{0, 1\}$  which requires circuits of size greater than  $\frac{N}{3 \log N}$ ,  $M$  is a machine which transforms concisely encoded circuits of size  $\frac{N}{3 \log N}$  into truth tables, and  $f(n) = n + 1$ . The associated function problem in **SAPEPP**, which could be called **SPARSE COMPLEXITY**, seeks the explicit construction of a hard Boolean function; a polynomial-time solution for this problem would imply, among other tectonic consequences, that  $\mathbf{P} = \mathbf{BPP}$  [13].

## 2 The Problems in PEPP

**EMPTY** is the following search problem: Given a circuit  $C$  with Boolean gates mapping  $[2^n - 1]$  to  $[2^n]$ , find a  $y \in [2^n]$  such that  $y \neq C(x)$  for all  $x \in [2^n - 1]$ .

► **Remark.** In this paper, we shall blur the distinction between bitstrings and binary integers. Our Boolean circuits have a domain and range whose cardinality is not necessarily a power of two, which may seem peculiar. In this paper we shall consider Boolean circuits mapping  $[M]$  to  $[N]$ , where  $M, N$  are arbitrary integers greater than one. Such a circuit  $C$  has  $\lceil \log M \rceil$  inputs and  $\lceil \log N \rceil$  outputs, and for all  $x$   $C(x)$  is defined to be  $C(M - 1)$  if  $x \geq M$ , and also for all  $x$   $C(x) = N - 1$  whenever the value computed by  $C$  on input  $x$  (or on input  $M - 1$  if  $x \geq M$ ) is at least  $N$ . Hence, **EMPTY** can be defined in terms of any circuit  $C : [M] \mapsto [M + 1]$  – or even  $C : [M] \mapsto [N]$  as long as  $M < N$ . For larger  $N$  the problem may be easier, but it is reducible to **EMPTY** (as long as  $\log N \leq \text{poly log } M$ ).

Coming back to **EMPTY**, we can now define a class of total functions **PEPP** as all total functions that are polynomial-time reducible to **EMPTY**. One rather immediate – and yet a little surprising – fact to observe about **PEPP** is the following:

► **Theorem 1.**  $\mathbf{FNP} \subseteq \mathbf{PEPP}$ .

**Proof.** We prove that **SAT** can be reduced to **EMPTY**. Let  $\phi$  be a CNF formula with  $n$  variables, without loss of generality not satisfied by the all-true truth assignment. Consider now the following polynomially computable function  $C$  from  $[2^n - 1]$  to  $[2^n]$ : For every truth assignment  $t$  different from the all-true one  $1^n$ ,  $C$  tests whether  $t$  satisfies  $\phi$ . If it does, then  $C(t) = 1^n$ , and if it does not then  $C(t) = t$ . Now, if we could solve **EMPTY**, that is, if we could find a solution  $s \in [2^n]$  not in the range of  $C$ , then we would have solved the **SAT** problem for  $\phi$ : If  $s \neq 1^n$  then  $\phi$  is satisfiable and  $s$  satisfies it; otherwise,  $\phi$  is unsatisfiable. ◀

This result suggests that **PEPP** is genuinely a subclass of  $\mathbf{TF}\Sigma_2$ , the generalization of **TFNP** to the first level of the polynomial hierarchy. Once we are dealing with  $\mathbf{TF}\Sigma_2$ , it is tempting to define classes such as **PEPP** as the set of all problems that can be reduced through  $\mathbf{FP}^{\mathbf{NP}}$  reductions – not just polynomial-time reductions – to a specific problem, such as **EMPTY** in the case of **PEPP**. This option becomes relevant when dealing with  $\alpha$ -**PEPP** in the next subsection.

As we sketched in the introduction, **EMPTY** and **PEPP** can be alternatively thought as a computationally constructive form of the union bound. A most prominent and early use of the union bound is in Shannon’s work on codes. The following problem motivated by Shannon’s construction has been recently identified: Given a code, which generically means a circuit  $C$  mapping  $[M]$  to  $[N]$  with  $N > M$ , find a bitstring  $x \in [N]$  whose Hamming distance from any codeword  $y$ , that is, any  $y$  such that  $y = C(z)$  for some  $z \in [M]$  is at least  $d$ , where  $d$  is the largest integer such that the Hamming ball of radius  $d - 1$  has fewer than  $N/M$  elements. This is known as the **REMOTE POINT** problem, studied extensively in Complexity and Cryptography [2, 3, 4].

► **Proposition 2.** **REMOTE POINT** is in **PEPP**.

**Proof.** Its proof of totality is an application of the union bound. ◀

In fact, **REMOTE POINT** is strictly speaking **PEPP**-complete, because any instance of **EMPTY** is also an instance of **REMOTE POINT** with  $d = 1$ .

Another natural problem lying in **PEPP** comes from the fact that graphs of bounded degree have logarithmic diameter. One way to capture this is through the problem **REMOTE VERTEX**: given a directed graph on  $[N]$  with vertices of outdegree at most 2, specified by circuits  $C_L, C_R : [N] \rightarrow [N]$  which output the “left” and “right” successors of a given node respectively, find a vertex whose distance from the all-zero vertex is at least  $\log N$ .

► **Proposition 3.** **REMOTE VERTEX** is in **PEPP**.

**Proof.** Consider the circuit that takes as input a string  $s \in \{L, R\}^*$  of length  $0 \leq |s| \leq \log N - 1$ , and outputs the vertex we arrive at by starting with the all-zero vertex and repeatedly applying  $C_L$  or  $C_R$  to the current input based on the next character in  $s$ . This circuit maps each path of length at most  $\log N - 1$  starting at the all-zero vertex to its endpoint, a vertex in  $[N]$ . As there are at most  $N - 1$  such paths, this is a valid instance of **EMPTY**, whose solution is indeed a remote vertex. ◀

One can define several variants of **REMOTE VERTEX** using other implicit representations of graphs, for example the representations for undirected and directed graphs used to define the canonical problems of bounded degree for **PPA** and **PPAD** [18]. Both of these variants reduce to the version of **REMOTE VERTEX** defined above.

Next we introduce two problems in **PEPP** capturing two other classical applications of the union bound. In  $\delta$ -**RIGID MATRIX COMPLETION**, where  $0 < \delta < \frac{1}{3}$ , we are given the first  $\lceil \log n \rceil$  rows of an  $n \times n$  matrix with elements in  $\mathbf{GF}_2$ . We seek to complete this to a full matrix in  $\mathbf{GF}_2^{n \times n}$  that is  $\delta$ -rigid: it cannot be turned into a matrix of rank  $\leq \delta n$  by changing  $n^\delta$  or fewer entries in each row.

Why do we have to phrase the quest for the rigid matrix as a completion problem? The reason is that the alternative (“Given  $n$ , find an  $n \times n$  rigid matrix”) is a *sparse* problem, that is, it has a polynomially (in  $n$ ) many instances of length  $\leq n$ , which places it in complexity limbo, see e.g. [16]; alternatively, if  $n$  is given in binary, then the problem is even more ill-posed since an exponentially long output is required<sup>2</sup>.

To see that  $\delta$ -**RIGID MATRIX COMPLETION** reduces to **EMPTY**, let  $N = 2^{n^2 - n \log n}$  denote the total number of possible completions of the matrix, and let  $M$  denote the number of pairs  $(L, S)$  where  $L$  is a  $n \times n$  matrix of rank at most  $\delta n$  and  $S$  is a matrix that has at

<sup>2</sup> A related question is, how large should be the given part of the matrix in order to avoid sparsity? Giving the first row is not enough, since there are, up to isomorphism,  $n + 1$  such rows, and a similar argument precludes finitely many rows. With  $\log n$  rows in the input, the problem is arguably no longer sparse.

most  $n^\delta$  ones per row. There are at most  $2^{2\delta n^2}$  choices for  $L$  and at most  $\binom{n}{n^\delta}^n < 2^{n^{1+\delta} \log n}$  choices for  $S$ , hence  $M < 2^{2\delta n^2 + n^{1+\delta} \log n}$ . Now consider the circuit  $C : [M] \mapsto [N]$  that takes an input in  $[M]$ , interprets it as the encoding of a pair  $(L, S)$ , computes the sum  $L + S$ , and outputs the element of  $[N]$  encoding this sum if it is a completion of the given matrix, or else it outputs an arbitrary element of  $[N]$ , for example the element representing the trivial matrix completion that sets all remaining entries of the given matrix to zero. Note that  $M < N$  as long as  $n$  is large enough that  $(1 - 2\delta)n^2 > (n^{1+\delta} + n) \log n$ . Any element of  $[N]$  not in the range of  $C$  must be the encoding of a matrix completion that cannot be expressed in the form  $L + S$ , hence is rigid.

RAMSEY-ERDŐS COMPLETION is the problem of finding an  $n$ -node graph with no independent set of size  $k = 4\lceil \log n \rceil$  and no clique of this size, given the connectivity of  $\ell = \lceil \log n \rceil$  nodes in the graph.

There are  $N = 2^{\binom{n-\ell}{2}}$  completions of the given graph. The completions containing a clique or independent set of size  $k$  are parameterized by tuples  $(A, b, x)$  where  $A$  is a vertex set of size  $k$ ,  $b$  is a bit indicating whether  $A$  forms a clique or independent set, and  $x$  is a bitstring indicating which edges belong to the completion, excluding those having both endpoints in  $A$  and those having one endpoint among the  $\ell$  vertices whose connectivity is given in the problem input. There are  $\binom{n}{k}$  possible values for  $A$ , 2 possible values for  $b$ , and at most  $2^{\binom{n-\ell}{2} - \binom{k-\ell}{2}}$  possible values for  $x$ , hence at most  $M = \binom{n}{k} 2^{1 + \binom{n-\ell}{2} - \binom{k-\ell}{2}}$  possible values for the triple  $(A, b, x)$ . When  $k = 4\ell$  and  $\ell = \lceil \log n \rceil$ , it follows from a standard calculation that  $M < N$ .

We have established this result:

► **Proposition 4.**  $\delta$ -RIGID MATRIX COMPLETION and RAMSEY-ERDŐS COMPLETION are in PEPP.

Of these problems RAMSEY-ERDŐS COMPLETION seems the easiest computationally, as it belongs in a variant of **BPP** in which  $n^{O(\log n)}$  computations are allowed.

## 2.1 The Problem COMPLEXITY

The field of Circuit Complexity is about identifying a Boolean function with  $v$  variables requiring a number of gates that grows faster than polynomially in  $v$ . It is well known since Shannon's union bound proof [20] that almost all Boolean functions with  $v$  variables have complexity at least  $2^{\frac{cv}{\log v}}$  for some  $c > 0$ ; however, no explicit function of complexity that is not  $O(v)$  is known.

We can now define COMPLEXITY: given a bitstring  $x$  of length  $n$ , find a Boolean function with  $v = \lceil \log n \rceil + 1$  inputs which cannot be computed by an  $x$ -oracle circuit with  $c \cdot \frac{n}{\log^2 n}$  gates, where  $c > 0$  is a fixed constant. Here, by " $x$ -oracle circuit" we mean a Boolean circuit which, besides the traditional AND, OR, NOT gates also has an ORACLE gate, with fan-in  $\lceil \log n \rceil$ , which when its inputs are the bits  $b_1, \dots, b_{\lceil \log n \rceil}$ , the value of the gate is the  $b+1$ -th bit of  $x$ , where  $b < n$  is the integer spelled by the bits.

We shall assume that, for each  $k, \ell > 0$ , we have a standard representation  $R^{k, \ell}$  of such oracle circuits, where  $k$  is the number of inputs to the circuit and  $\ell$  is the fan-in of the oracle gates.  $R^{k, \ell}$  is a partial function (that is, possibly undefined) from bitstrings to circuits, such that:

- Every  $x$ -oracle circuit  $K$  has at least one bitstring  $z$  such that  $R^{k, \ell}(z) = K$ .
- Given  $z$ ,  $K = R^{k, \ell}(z)$  can be decoded in polynomial time.
- If  $K$  has  $g$  gates, the length of all  $z$ 's such that  $R^{k, \ell}(z) = K$  is at most  $c \cdot g \log^2 g$ , where  $c$  is a constant.

It is easy to see that these desiderata are satisfied by several standard and natural representations (for example, encoding every bit by two bits to create delimiters, encoding gate names by binary integers  $\leq g$  and similarly with gate types, and finally encoding the adjacency lists of the circuit graph). The extra  $\log g$  in the last item is due to the oracle gate, whose adjacency list requires  $\log^2 g$  bits.

Coming back to **COMPLEXITY**, it is, evidently, a computational problem that captures certain aspects of Complexity Theory. We shall show that it is a total problem, and in fact one in the class **PEPP**.

The argument is essentially Shannon's: given input  $x$  of length  $n$ , we construct a circuit  $C_x$  implementing the following polynomial-time (in  $n$ ) algorithm: on any input  $y$  also of length  $n$ ,  $C_x$  interprets  $y$  as a binary representation of an  $x$ -oracle circuit  $K_y = R^{k,\ell}(y)$  with  $k = \lfloor \log n \rfloor + 1$  input gates and fan-in  $\ell = \lfloor \log n \rfloor$ , and goes on to construct it (if  $R^{k,\ell}(y)$  is undefined,  $C_x$  outputs a default string). Next,  $C_x$  simulates  $K_y$  consecutively on each possible input in  $[2^{\lfloor \log n \rfloor + 1}]$ . The output of  $C_x$  is then the concatenation of these  $2^{\lfloor \log n \rfloor + 1}$  bits output by the circuit  $K_y$ , in the order in which they were produced.

In other words, the circuit  $C_x$  maps  $M = [2^n]$  (all inputs  $y$  of length  $n$ ) to  $N = 2^{2^{\lfloor \log n \rfloor + 1}}$  possible outputs, and it is clear that  $N > M$ . Therefore, if we were able to solve **EMPTY** and obtain a possible output not realized by any possible input, we would be able to find the table of a Boolean function with  $\lfloor \log n \rfloor + 1$  inputs which cannot be represented by  $n$  bits, and therefore requires  $\Omega(\frac{n}{\log^2 n})$  gates. This completes the proof of the following result:

► **Proposition 5.** *COMPLEXITY is in PEPP.*

## 2.2 Wasteful counting and $\alpha$ -PEPP

When the union bound is used to prove the existence of objects with a certain property by showing that a random object satisfies the property with positive probability, this success probability is typically not exponentially small. The reason is that this genre of existence proof seems inherently wasteful: the union bound adds probabilities of events that typically overlap, while counting objects such as non-rigid matrices and circuits typically counts the same object many times (for example, all permutations of gate names), and there seems to be no way to be accurate enough. To capture the complexity of the search problems implied by union bound arguments with a significant “margin of error”, we define a family of complexity classes  $\alpha$ -**PEPP**, parameterized by a function  $\alpha : \mathbb{N} \rightarrow \mathbb{R}_+$ .

The complexity class  $\alpha$ -**PEPP** is defined to consist of all total functions that are polynomial-time reducible to the following  $\alpha$ -**EMPTY** problem. An instance of  $\alpha$ -**EMPTY** is given by a bitstring of length  $n$  interpreted as a description of a circuit  $C$  mapping  $[M]$  to  $[N]$ , where  $N/M > 1 + \alpha$ . The search problem is to find  $y \in [N]$  such that  $y \neq C(x)$  for all  $x \in [M]$ .

Note that **0-PEPP** = **PEPP**. The complexity class **1-PEPP**, which we will denote by **APEPP** in the sequel, contains most of the search problems introduced earlier in this section.

► **Proposition 6.** *The problems  $\delta$ -RIGID MATRIX COMPLETION, RAMSEY-ERDŐS COMPLETION, and COMPLEXITY all belong to APEPP.*

**Proof.** Above, we presented reductions from  $\delta$ -RIGID MATRIX COMPLETION, RAMSEY-ERDŐS COMPLETION, and COMPLEXITY to **EMPTY** with the following parameters.

- For a  $\delta$ -RIGID MATRIX COMPLETION instance of size  $n \times n$ , the reduction yields a circuit of size  $\text{poly}(n)$  mapping  $[M]$  to  $[N]$ , where  $M = 2^{2\delta n^2 + n^{1+\delta} \log n}$  and  $N = 2^{n^2 - n \log n}$ ,

$$N/M = 2^{(1-2\delta)n^2 - (n^{1+\delta} + n) \log n}.$$

The ratio  $N/M$  exceeds 2 when  $\delta < \frac{1}{3}$ ,  $n > 125$ .

- For a RAMSEY-ERDŐS COMPLETION instance with  $n$  vertices, the reduction yields a circuit of size  $\text{poly}(n)$  mapping  $[M]$  to  $[N]$ , where  $M = \binom{n}{4^{\lceil \log n \rceil}} 2^{1 + \binom{n - \lceil \log n \rceil}{2} - \binom{3^{\lceil \log n \rceil}}{2}}$ ,  $N = 2^{\binom{n - \lceil \log n \rceil}{2}}$ ,

$$\frac{N}{M} = \frac{2^{\binom{3^{\lceil \log n \rceil}}{2} - 1}}{\binom{n}{4^{\lceil \log n \rceil}}}.$$

The ratio  $N/M$  exceeds 2 when  $n > 8$ .

- For a COMPLEXITY instance of length  $n$ , the reduction yields a circuit of size  $\text{poly}(n)$  mapping  $[M]$  to  $[N]$ , where  $M = 2^n$ ,  $N = 2^{2^{\lceil \log n \rceil + 1}}$ ,

$$N/M = 2^{2^{\lceil \log n \rceil + 1} - n}.$$

The ratio  $N/M$  is greater than 2 for all  $n \in \mathbb{N}$ .<sup>3</sup>

Thus, the reductions presented earlier verify that all three problems belong to **APEPP**. ◀

It seems unlikely that **APEPP** contains a **PEPP**-complete problem, due to the following upper bound on the complexity of **APEPP**.

► **Theorem 7.** *For any function  $\alpha(n) > \frac{1}{\text{poly}(n)}$ , we have  $\alpha\text{-PEPP} \subseteq \mathbf{FZPP}^{\mathbf{NP}} \subseteq \mathbf{FP}^{\mathbf{NP}}|_{\text{poly}}$  where **FZPP** denotes functional **ZPP**.*

**Proof.** First we show  $\alpha\text{-EMPTY} \in \mathbf{FZPP}^{\mathbf{NP}}$ . Consider a circuit  $C$  mapping  $[M]$  to  $[N]$ , where  $N/M > 1 + \alpha$ . Let  $R(C)$  denote the range of  $C$ , i.e. the set of all  $y \in [N]$  such that there exists  $x \in [M]$  with  $C(x) = y$ . The probability that a random  $y \in [N]$  belongs to  $R(C)$  is at most  $M/N < 1/(1 + \alpha)$ . Hence, if  $k = \lceil n/\alpha \rceil$  and  $y_1, y_2, \dots, y_k$  are independent random elements of  $[N]$ , the probability that  $\{y_1, \dots, y_k\} \subseteq R(C)$  is less than  $(1 + \alpha)^{-k} < e^{-n}$ . Consider an algorithm that randomly samples the  $y_i$ , and queries an **NP** oracle whether there exists  $x \in [M]$  such that  $C(x) = y_i$ , and outputs the first  $y_i$  for which the oracle confirms no such  $x$  exists: this shows  $\alpha\text{-EMPTY} \in \mathbf{FZPP}^{\mathbf{NP}}$ .

To see the second inclusion of the theorem, suppose an algorithm that uses randomness  $r$ ,  $|r| \leq \text{poly}(n)$  fails with probability  $< e^{-n}$  for any length  $n$  input. By the union bound, a random sample of  $r$  has positive probability of containing a valid solution to *every* length- $n$  input instance. In fact, this probability is greater than  $1 - 2^n \cdot e^{-n}$ . Hence, there exists an advice string such that for every length- $n$  instance, the algorithm finds a correct output. ◀

We conclude this section by showing a collapse of the complexity classes  $\alpha\text{-PEPP}$  for  $\frac{1}{\text{poly}(n)} \leq \alpha(n) \leq 2^{\text{poly}(n)}$  under  $\mathbf{FP}^{\mathbf{NP}}$  reductions.

<sup>3</sup> Actually, the ratio is greater than or equal to 2 for all  $n$ , but it is equals 2 when  $n + 1$  is a power of 2. Since the definition of  $\alpha\text{-EMPTY}$  requires the strict inequality  $N/M > 1 + \alpha$ , we need to correct for this technicality with a small modification in the definition of COMPLEXITY, tweaking the problem definition to use a slightly smaller constant  $c$  so that all circuits of the appropriate size or less can be encoded in  $n - 1$  bits, instead of  $n$ .

## 44:10 Total Functions in the Polynomial Hierarchy

► **Theorem 8.** *If  $\frac{1}{\text{poly}(n)} \leq \alpha(n) \leq 2^{\text{poly}(n)}$ , then  $\alpha$ -PEPP and APEPP are equivalent under  $\mathbf{FP}^{\mathbf{NP}}$  reductions.*

**Proof.** For any positive integers  $N, k$ , let  $T : [N^k] \rightarrow [N]^k$  denote the function that takes the binary representation of a number  $x \in [N^k]$ , writes  $x$  in base  $N$  as a sequence of  $k$  digits (each an element of  $[N]$ ), and outputs the binary string obtained by concatenating the binary representations of each of these  $k$  base- $N$  digits.

Suppose  $\beta(n), \gamma(n) : \mathbb{N} \rightarrow \mathbb{R}_+$  are any two functions such that

$$k(n) \triangleq \lceil \log_{1+\beta(n)}(1 + \gamma(n)) \rceil \leq \text{poly}(n).$$

We can reduce  $\beta$ -EMPTY to  $\gamma$ -EMPTY as follows. Given a length- $n$  bitstring describing a circuit that computes a function  $C : [M] \rightarrow [N]$ , let  $k = k(n)$  and construct the description of a circuit that computes the function  $C' : [M^k] \rightarrow [N^k]$  defined via the following composition:

$$[M^k] \xrightarrow{T} [M]^k \xrightarrow{C^k} [N]^k \xrightarrow{T^{-1}} [N^k].$$

Here  $C^k$  denotes the function that applies  $C$  to each element of a  $k$ -tuple.

Assuming  $N/M > 1 + \beta(n)$ , we have  $N^k/M^k > (1 + \beta(n))^k \geq 1 + \gamma(n)$ , by the definition of  $k$ . Hence, by solving an instance of  $\gamma$ -EMPTY and applying the function  $T$ , we obtain a  $k$ -tuple  $(y_1, \dots, y_k)$  that is not in the range of the function  $C^k : [M]^k \rightarrow [N]^k$ . Now, given an  $\mathbf{NP}$  oracle, we can proceed as in the proof of Theorem 7 to find  $y \in [N]$  such that for all  $x \in [M]$ ,  $y \neq C(x)$ . Namely, for  $1 \leq i \leq k$  one queries the  $\mathbf{NP}$  oracle to find out if there exists some  $x_i \in [M]$  such that  $C(x_i) = y_i$ . If such an  $x_i$  existed for each  $i$ , then  $C^k(x_1, \dots, x_k)$  would equal  $(y_1, \dots, y_k)$  contradicting our assumption that  $(y_1, \dots, y_k)$  is not in the range of  $C^k$ . Therefore, for at least one value of  $i$  the oracle will answer that no  $x_i \in [M]$  satisfies  $C(x_i) = y_i$ , and we can output this  $y_i$  as a solution of the given  $\beta$ -EMPTY instance.

We have shown a  $\mathbf{FP}^{\mathbf{NP}}$  reduction from  $\beta$ -PEPP to  $\gamma$ -PEPP whenever  $\log_{1+\beta(n)}(1 + \gamma(n)) = \text{poly}(n)$ . If  $\frac{1}{\text{poly}(n)} \leq \alpha(n) \leq 2^{\text{poly}(n)}$ , then a reduction from  $\alpha$ -PEPP to APEPP is obtained by taking  $\beta = \alpha$  and  $\gamma \equiv 1$ , and a reduction from APEPP to  $\alpha$ -PEPP is obtained by taking  $\beta \equiv 1$  and  $\gamma = \alpha$ . ◀

### 3 The SHATTERING Problem

We recall the definition of *shattering*, an important notion in finite set theory and classical learning theory:

► **Definition 9.** *A family of sets over some finite universe,  $F = \{s_1, s_2, \dots\}$ , shatters a set  $s$  if for every subset  $t \subseteq s$ , there exists  $s_i \in F$  such that  $t = s \cap s_i$ .*

The famous Sauer-Shelah lemma guarantees shattering properties if the family is large enough. Here it is stated in its “strong” form:

► **Theorem 10** (Sauer-Shelah Lemma, Strong). *A family  $F$  of finite sets shatters at least  $|F|$  sets.*

The more well-known statement of the Sauer-Shelah lemma is the weak form, which follows from the above:

► **Corollary 11** (Sauer-Shelah Lemma, Weak). *If a family of sets  $F$  over a universe of  $n$  elements satisfies  $|F| > \sum_{i=0}^{d-1} \binom{n}{i}$ , then  $F$  must shatter a set of cardinality at least  $d$ .*

**Proof.** (Of the weak form from the strong form:) There are at most  $\sum_{i=0}^{d-1} \binom{n}{i}$  sets in an  $n$ -element universe that have size less than  $d$ . ◀



It is natural to consider the search problem resulting from this lemma: given a family of sets over  $n$  elements, which can be represented as  $n$  bit strings, find a large shattered set. This search problem is interesting for two reasons: first, its standard proof uses a counting argument that is, in essence, non-constructive, and second, it involves multiple alternations: given a family find the set (*exists*) such that *for all* subsets there *exists* a corresponding set in the family. In fact, this has one more alternation than all the problems we have considered previously, which belong in  $\mathbf{TF}\Sigma_2$ . Instead, this belongs in  $\mathbf{TF}\Sigma_3$ .

► **Definition 12.** Let  $\text{BinomSum}(n, d)$  denote  $\sum_{i=0}^{d-1} \binom{n}{i}$ .

► **Definition 13.** In the SHATTERING problem, we are given as inputs parameters  $n$ ,  $d$ , and  $k > \log(\text{BinomSum}(n, d))$ , and a circuit computing a function  $C : \{0, 1\}^k \rightarrow \{0, 1\}^n$ , representing  $2^k$  indexed sets the collection of which we will denote  $F$ . The search problem is to output either a pair of indices  $x_1 \neq x_2$  such that  $C(x_1) = C(x_2)$  (a collision, in which case the premise of the Sauer-Shelah lemma is not satisfied), or a subset  $Y \subseteq [n]$  of size  $|Y| = d$  that is shattered by the  $F$ , the range of  $C$ .

The following is now clear:

► **Proposition 14.** SHATTERING is in  $\mathbf{TF}\Sigma_3$ .

**Proof.** Consider the Turing Machine  $M((n, d, k, C), s, (u, i))$ , which:

1. checks that  $k > \log(\text{BinomSum}(n, d))$ ;
2. checks whether  $s$  is a string representing a tuple  $x_1, x_2$  of  $k$ -bit strings, in which case it accepts if  $C(x_1) = C(x_2)$  and rejects otherwise;
3. checks whether  $s$  is an  $n$ -bit string, in which case it accepts if  $s \cap u = s \cap C(i)$  and rejects otherwise.

Clearly,  $s$  solves SHATTERING on the input  $(n, d, k, C)$  when the conditions of the Sauer-Shelah lemma are not satisfied, or, if  $\forall u \exists i$  s.t.  $M((n, d, k, C), s, (u, i)) = 1$ . That SHATTERING is total is a consequence of the Sauer-Shelah lemma. ◀

More interestingly, we can place SHATTERING in a generalization of  $\mathbf{PPP}$  that lies within  $\mathbf{TF}\Sigma_3$ :

► **Theorem 15.** SHATTERING is in  $\mathbf{PPP}^{\Sigma_2}$

The main technical result is the following lemma, from which the theorem follows naturally.

► **Lemma 16.** Using a  $\Sigma_2$  oracle, one can compute a polynomial time function  $M$  mapping distinct sets in  $F$  to distinct sets shattered by  $F$ .

**Proof (Lemma 16, informal).**  $M$  is defined recursively on the size of  $F$ . For collections of size  $|F| = 1$ , the single element of  $F$  is mapped to the empty set which is certainly shattered by  $F$ .

Assume now that  $M(F')$  is defined for all collections  $F'$  of size  $|F'| < |F|$  (i.e.  $M$  defined for collection of size  $1, \dots, |F| - 1$ ). We show how to define  $M$  on  $F$ , first with an informal argument.

Suppose we have identified an element  $x$  that is in *at least one but not all* sets of  $F$ . Then we can write  $F = F_0 \cup F_1$ , dividing  $F$  into collections  $F_0$  of sets containing  $x$ , and  $F_1$  of sets that do not contain  $x$ .

Since  $|F_0|, |F_1| < |F|$ , by induction there exists  $M_0 : F_0 \rightarrow \{0, 1\}^n$  and  $M_1 : F_1 \rightarrow \{0, 1\}^n$  mapping each subcollection to sets shattered by that subcollection. Define  $M : F \rightarrow \{0, 1\}^n$  as follows:

## 44:12 Total Functions in the Polynomial Hierarchy

1. For  $s \in F_1$ , reuse the shattered set, i.e. let  $F(s) = F_1(s)$ .
2. For  $s \in F_0$ , if  $\forall s' \in F_1, M_1(s') \neq M_0(s)$ , we reuse the label for  $s$ , i.e.  $M(s) = M_0(s)$ . If  $\exists s' \in F_1$  such that  $M_1(s') = M_0(s)$ , then it must be that  $F$  shatters both  $M_0(s)$  and  $M_0(s) \cup x$ . Hence we can assign  $M(s) = M_0(s) \cup x$ .

In the informal construction above, we assumed that  $x$  is in some but not all sets of  $F$ . To define  $M$  consistently, we go through all elements in the universe  $\{0, 1, \dots, n-1\}$  in order, and we divide  $F$  into those sets that contain the element and those that don't – since these sets are  $n$ -bit strings, we divide them into those strings with 1 in the first coordinate, and those with 0 in first coordinate). It is possible that one side is empty and the other is all of  $F$ ; in this case, we continue splitting by containment of subsequent elements. When one side is empty, then all labels assigned to sets in the non-empty side are reused. This gives a way to build a complete binary tree of subfamilies starting with  $F$  at level 0, and where the  $i+1$ -st level comes from splitting the previous level by containment of element  $i$ .  $M$  then is built recursively from the leaves up. ◀

**Proof (Lemma 16).** We describe how to compute  $M(s)$  for a given  $s \in F$ . Define  $T_n$  to be the labeled binary tree with  $2^n$  leaves representing  $n$ -bit strings; level  $i$  contains nodes labelled by the  $2^i$  binary strings of length  $i$ , and the children of a node labelled with  $s \in \{0, 1\}^i$  is  $s \cdot 0$  and  $s \cdot 1$ .

The idea is to go up  $T_n$ , beginning from the leaf representing set  $s$ , computing  $y^{(n)}, y^{(n-1)}, \dots, y^1$ . This path is unique and has length  $n$ ; denote this path with  $P$ , and its nodes as  $P(n), \dots, P(0)$ , from leaf to root (we will interchangeably use  $P(i)$  to refer to both a node in  $T_n$ , and its associated label, a string of length  $i$ ).

1. When we begin at node  $P(n)$  we initialize  $y$  with the empty set label  $y^n = 0^n$ .
2. Assume we have traversed  $T_n$  up to level  $i$ , i.e.  $P(i)$ .
3. If the node  $P(i)$  is the right child of  $P(i-1)$ , move up to  $P(i-1)$  with  $y^{(i-1)} := y^i$ .
4. For  $P(i)$  that is the left child of  $P(i-1)$ , denote the right child of  $P(i-1)$  (sibling of  $P(i)$ ) and its label as  $P'(i)$ , and denote by  $F_{P'(i)}$  the subcollection of sets in  $F$  that have the label  $P'(i)$  as its prefix (i.e., those sets that agree on the inclusion/exclusion decisions of the first  $i$  elements represented by node  $P'(i)$ ). We check whether  $F_{P'(i)}$  also shatters  $y^i$ , in which case we reuse  $y^i$  but flipping bit  $i$  to 1, i.e.  $y^{(i-1)} := y^i; y_i^{(i-1)} := 1$ . Whether  $F_{P'(i)}$  (or any particular subfamily corresponding to a node in  $T_n$ ) shatters  $y$  can be established in  $O(i)$  time: the algorithm checks whether  $\forall z \in \{0, 1\}^n \exists w \in \{0, 1\}^k (C(w) \in F_{P'(i)}) \wedge (z \cap y = C(w) \cap y)$ . This can be determined with one call to the  $\Sigma_2$  oracle. Note that  $C(w) \in F_{P'(i)}$  can be represented as an  $\wedge$  of  $i$  equalities.

The following invariant is maintained throughout the algorithm: after completing level  $i$ ,  $F_{P(i)}$  shatters  $y^i$ . This is clearly true at level  $n$ . With level  $i$  completed, if the algorithm assigns  $y^{i-1} = y^i$ , the invariant is maintained as  $y$  does not change. The only way  $y^{i-1}$  changes is if  $y_1^{i-1} = 1$ ; this implies both  $F_{P(i)}$  and  $F_{P'(i)}$  shattered  $y^i$ .

If  $|F| = 1$  (the range of  $C$  is one set), assigning the empty set to the lone element is correct. For  $s \neq s'$  in the range of  $C$ , let  $P(i)$  be their lowest common ancestor in level  $i < n$ . Denote its children as  $P(i+1)$  and  $P'(i+1)$ ; without loss of generality,  $P(i)$  is on the path for  $s$  and  $P'(i)$  is on the path for  $s'$ . Consider the algorithm at level  $i+1$  when run on both inputs to compute the shattered sets  $y$  and  $y'$ : if  $y^{(i+1)} \neq y'^{(i+1)}$ , they will remain different for the remainder of the algorithm since at level  $j$  we can only change the  $j$ -th bit. If  $y^{(i+1)} = y'^{(i+1)}$ , then both  $P(i+1)$  and  $P'(i+1)$  shatter  $y^{(i+1)}$  so the algorithm will set  $y^i \neq y'^i$ . ◀

**Proof (Lemma 16  $\implies$  Theorem).** Given an instance  $C$  of SHATTERING, we shall describe an instance  $H$  of PIGEONHOLE $^{\Sigma_2}$  – that is, a hashing circuit with  $k$  input gates and  $2^k - 1$  possible outputs, whose computation makes calls to a  $\Sigma_2$  oracle – which solves this instance. First, the circuit  $H$  determines through an oracle call if  $C$  has a collision, and, if it does – two strings  $x, y \in [2^k]$  such that  $x > y$  and  $C(x) = C(y)$  – it computes a perturbation of the identity permutation on  $[2^k]$  which exposes the collision:  $H$  maps  $x$  to  $y$ , if  $x \neq 2^k - 1$  it maps  $2^k - 1$  to  $x$ , and  $H$  is the identity on all other strings.

If  $C$  has no collision, then on input  $x \in [2^k]$   $H$  first computes the distinct set  $C(x)$  and then implements the lemma to compute the corresponding set  $M(C(x))$  shattered by the  $C$  family of sets. If the set  $M(C(x))$  is smaller than  $d$ , the computation ends here and the set is output, in a representation which encodes subsets of  $[n]$  in order of increasing size; since by assumption  $2^k - 1 \geq \text{BinomSum}(n, d)$ , any set smaller than  $d$  can be represented. If the set is of size  $d$  or larger, then the first  $d - 1$  elements of the set are output. This completes the reduction.  $\blacktriangleleft$

It turns out that  $\text{TF}\Sigma_3$  contains another natural problem aside from SHATTERING, based on a simple fact in graph theory dating back to the 1950's [15].

► **Definition 17.** A vertex  $v$  in a digraph  $G$  is called a king if every vertex can be reached from  $v$  by a path of length at most 2.

► **Definition 18.** A digraph  $G$  is called a tournament if for every pair of distinct vertices  $u, v \in G$ , exactly one of the directed edges  $(u, v)$  or  $(v, u)$  is present in  $G$ .

► **Lemma 19.** Every tournament has a king [15].

**Proof.** Given a vertex  $v$ , will say that the *court* of  $v$  is the set of vertices reachable from  $v$  in exactly one step, and the *domain* of  $v$  is the set of vertices reachable from  $v$  in exactly 1 or 2 steps. By definition, a king is a vertex whose domain contains all other vertices. Starting with an arbitrary vertex  $v$ , we can now locate a king as follows: if the domain of  $v$  contains all other vertices then  $v$  is a king and we are done. Otherwise there exists a  $u \neq v$  outside the domain of  $v$ , and we continue our search from  $u$ . To see that this iterative process terminates, note that at each step the size of the court of our current vertex strictly increases: if  $u$  is outside the domain of  $v$ ,  $u$ 's court must contain at least every element of  $v$ 's court, and must also contain  $v$ .  $\blacktriangleleft$

This gives rise to the following total search problem KING: given a tournament, represented as a circuit  $C : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$  determining if the directed edge  $(u, v)$  is present, find either a king or a pair of inputs  $x \neq y$  such that  $C(x, y) = C(y, x)$  (proving that  $C$  does not define a tournament). Since KING seeks a vertex  $k$  such that for all other vertices  $v$  there is an intermediate vertex  $i$  which is reachable from  $k$  in one or zero steps, and such that  $(i, v)$  is an edge, this alternation of quantifiers implies the following:

► **Proposition 20.** KING is in  $\text{TF}\Sigma_3$ .

The proof of Lemma 19 showing the totality of KING is eerily reminiscent of familiar totality arguments for the class PLS. In PLS, the input is an implicitly defined directed graph, along with a polynomial time “potential function” mapping vertices to natural numbers, such that the potential is strictly increasing along edges. The goal is to find a vertex with no outgoing edges, and the potential function gives us a syntactic guarantee that the underlying graph is acyclic and therefore that such a vertex must exist. In the case of KING, we have a similar situation. Given a circuit defining a tournament, we can generate another implicitly

defined graph which has a directed edge  $(v, u)$  if and only if  $u$  is outside the domain of  $v$ , and a potential function which assigns each vertex a potential equal to the size of its court. A vertex with no outgoing edges in this graph will be a king, and since the potential increases along edges, such a vertex must exist. If both the “edge function” (the function finding an outgoing edge of a vertex or telling us that none exists) and the potential function were computable in  $\Sigma_2$ , this would place KING in  $\mathbf{PLS}^{\Sigma_2}$ . However, it seems that only the edge function has this property: computing the potential requires solving a rather generic counting problem. This leaves us in a curious situation, where the proof that a solution exists uses an implicit potential function, but directly computing the potential is seemingly harder than finding a solution. We do not know of another natural total function with this property.

## 4 Discussion and Open Problems

We have introduced a polynomial hierarchy of total functions, whose first couple of levels are populated with interesting computational problems and complexity subclasses with intriguing structural properties. Naturally, a host of questions remain:

- Does the total function hierarchy behave in similar ways as the polynomial hierarchy – for example, does it collapse upwards? As we have mentioned, the answer to this question is already known, modulo relativization, and it is negative: there are oracles with respect to which  $\mathbf{TFNP} = \mathbf{FP}$  and yet  $\mathbf{TF}\Sigma_2 \neq \mathbf{FP}^{\mathbf{NP}}$  [7]. We have not explored how this result extends to higher levels. After a preprint of this article was posted online, Ofer Grossman showed us a sketch of an argument that, if  $\mathbf{TFNP} = \mathbf{TF}\Sigma_2$ , then the decisional polynomial hierarchy does collapse (Ofer Grossman, personal communication). This can be shown to imply that the total function hierarchy collapses.
- A very striking apparent difference between  $\mathbf{TFNP}$  and  $\mathbf{TF}\Sigma_2$  is the dearth of diversity in the latter. There are half a dozen apparently distinct complexity subclasses of  $\mathbf{TFNP}$ , corresponding to natural genres of existence proofs. In contrast, in  $\mathbf{TF}\Sigma_2$  we have identified  $\mathbf{PPEP}$ , but – despite some intense daydreaming – no other credible class. For example, recall that  $\mathbf{PLS}$  is the class of all problems in  $\mathbf{TFNP}$  reducible to SINK: “Given the circuit representation of a DAG, find a sink” (details of the representation omitted). It is natural to ask – and we did: “How about the problem SOURCE? It is in  $\mathbf{TF}\Sigma_2$ , of course, but does it define its own class?” It turns out that SOURCE is in  $\mathbf{PEPP}$ ...  
For  $\mathbf{TFNP}$ , the invention of new natural subclasses is impeded by the result in [11], establishing, through Herbrand’s Theorem, that any such subclass capturing a style of existence proofs in first-order logic must correspond to a *finitary* property of first-order structures: one that is false for infinite structures. How about the logic formulae corresponding to  $\mathbf{TF}\Sigma_2$ ? These would be the so-called Schönfinkel-Bernays formulae (first-order formulae preceded by a sequence of quantifiers of the form  $\forall^*\exists^*$ ), a much studied class in Logic but also in Complexity (it had been known since the 1930s that this is a decidable class). Is there a result restricting the usefulness of such formulae in characterizing total search problems, analogous to – but perhaps stricter than – Herbrand’s theorem for existentially quantified (Herbrand) formulas?
- Is COMPLEXITY complete for  $\mathbf{APEPP}$  under  $\mathbf{P}^{\mathbf{NP}}$  reductions? This would be a tremendously interesting result. Naturally, any finer completeness result for COMPLEXITY would be even more exciting.
- Is REMOTE POINT with large  $d$  complete for  $\mathbf{APEPP}$  under  $\mathbf{P}^{\mathbf{NP}}$  reductions? That would be very interesting as well – especially if it holds true even in the special case in which the code is *linear*.

- The **SAPEPP** class, as defined in the related work subsection, encompasses some of the most important problems in Complexity Theory. Sparsity complicates proving these problems intractable, and yet we know already some fascinating reductions between them. Does **SAPEPP** have natural complete problems? Is **SPARSE COMPLEXITY** complete for it?
- The problem **KING** encompasses a novel aspect of total functions related to local optimality. Problems in the class **PLS** are presented in terms of an implicit DAG defined in terms of an edge function and a potential function. Higher in the hierarchy, **KING** is defined only in terms of an edge function, while the DAG property is established through an *extraneous proof*, that is, a proof not encoded in the instance's description in terms of an explicit potential function. Unless  $\#P$  is in the polynomial hierarchy, **KING** does not appear to belong in  $\mathbf{PLS}^{\Sigma_2}$ : it is a problem in  $\mathbf{TF}\Sigma_3$  whose totality follows from a local optimality argument, and yet one that is *sui generis*, in a class in and by itself. Are there such problems in **TFNP**?

---

## References

- 1 Leonard Adleman. Two theorems on random polynomial time. In *19th Annual Symposium on Foundations of Computer Science*, pages 75–83, 1978. doi:10.1109/SFCS.1978.37.
- 2 Noga Alon, Rina Panigrahy, and Sergey Yekhanin. Deterministic approximation algorithms for the nearest codeword problem. In Irit Dinur, Klaus Jansen, Joseph Naor, and José Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 339–351, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.
- 3 Vikraman Arvind and Srikanth Srinivasan. Circuit lower bounds, help functions, and the remote point problem, 2009. arXiv:0911.4337.
- 4 Vikraman Arvind and Srikanth Srinivasan. The remote point problem, small bias space, and expanding generator sets. In *27th International Symposium on Theoretical Aspects of Computer Science - STACS 2010*, pages 59–70, 2010.
- 5 Frank Ban, Kamal Jain, Christos H. Papadimitriou, Christos-Alexandros Psomas, and Aviad Rubinfeld. Reductions in PPP. *Inf. Process. Lett.*, 145:48–52, 2019. doi:10.1016/j.ipl.2018.12.009.
- 6 Boaz Barak, Anup Rao, Ronen Shaltiel, and Avi Wigderson. 2-source dispersers for sub-polynomial entropy and ramsey graphs beating the frankl-wilson construction. In *Proceedings of the Thirty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '06, page 671–680, New York, NY, USA, 2006. Association for Computing Machinery. doi:10.1145/1132516.1132611.
- 7 Harry Buhrman, Lance Fortnow, Michal Koucký, John D. Rogers, and Nikolai K. Vereshchagin. Does the polynomial hierarchy collapse if onto functions are invertible? *Theory Comput. Syst.*, 46(1):143–156, 2010. doi:10.1007/s00224-008-9160-8.
- 8 Gil Cohen. *Two-Source Dispersers for Polylogarithmic Entropy and Improved Ramsey Graphs*, page 278–284. Association for Computing Machinery, New York, NY, USA, 2016. doi:10.1145/2897518.2897530.
- 9 Constantinos Daskalakis, Paul W. Goldberg, and Christos H. Papadimitriou. The complexity of computing a nash equilibrium. *SIAM J. Comput.*, 39(1):195–259, 2009. doi:10.1137/070699652.
- 10 Aris Filos-Ratsikas and Paul W. Goldberg. Consensus halving is ppa-complete. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 51–64. ACM, 2018. doi:10.1145/3188745.3188880.

- 11 Paul W. Goldberg and Christos H. Papadimitriou. Towards a unified complexity theory of total functions. In Anna R. Karlin, editor, *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, volume 94 of *LIPIcs*, pages 37:1–37:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.ITCS.2018.37.
- 12 Pavel Hubáček, Moni Naor, and Eylon Yogev. The journey from NP to TFNP hardness. In Christos H. Papadimitriou, editor, *8th Innovations in Theoretical Computer Science Conference, ITCS 2017, January 9-11, 2017, Berkeley, CA, USA*, volume 67 of *LIPIcs*, pages 60:1–60:21. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPIcs.ITCS.2017.60.
- 13 Russell Impagliazzo and Avi Wigderson. P = BPP if E requires exponential circuits: Derandomizing the XOR lemma. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing, STOC '97*, page 220–229, New York, NY, USA, 1997. Association for Computing Machinery. doi:10.1145/258533.258590.
- 14 Ilan Komargodski, Moni Naor, and Eylon Yogev. White-box vs. black-box complexity of search problems: Ramsey and graph property testing. In Chris Umans, editor, *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 622–632. IEEE Computer Society, 2017. doi:10.1109/FOCS.2017.63.
- 15 H. G. Landau. On dominance relations and the structure of animal societies: III The condition for a score structure. *The bulletin of mathematical biophysics*, 15(2):143–148, June 1953. doi:10.1007/BF02476378.
- 16 Stephen R. Mahaney. Sparse complete sets for NP: solution of a conjecture of berman and hartmanis. In *21st Annual Symposium on Foundations of Computer Science, Syracuse, New York, USA, 13-15 October 1980*, pages 54–60. IEEE Computer Society, 1980. doi:10.1109/SFCS.1980.40.
- 17 Noam Nisan and Avi Wigderson. Hardness vs randomness. *Journal of Computer and System Sciences*, 49(2):149–167, 1994. doi:10.1016/S0022-0000(05)80043-1.
- 18 Christos H. Papadimitriou. On the complexity of the parity argument and other inefficient proofs of existence. *Journal of Computer and System Sciences*, 48(3):498–532, 1994. doi:10.1016/S0022-0000(05)80063-7.
- 19 N. Sauer. On the density of families of sets. *J. Combinatorial Theory Ser. A*, 13:145–147, 1972. doi:10.1016/0097-3165(72)90019-2.
- 20 C. E. Shannon. The synthesis of two-terminal switching circuits. *The Bell System Technical Journal*, 28(1):59–98, 1949.
- 21 Saharon Shelah. A combinatorial problem; stability and order for models and theories in infinitary languages. *Pacific J. Math.*, 41:247–261, 1972. URL: <http://projecteuclid.org/euclid.pjm/1102968432>.
- 22 Katerina Sotiraki, Manolis Zampetakis, and Giorgos Zirdelis. Ppp-completeness with connections to cryptography. In Mikkel Thorup, editor, *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 148–158. IEEE Computer Society, 2018. doi:10.1109/FOCS.2018.00023.
- 23 Luca Trevisan. Extractors and pseudorandom generators. *J. ACM*, 48(4):860–879, July 2001. doi:10.1145/502090.502099.
- 24 Leslie G. Valiant. Graph-theoretic arguments in low-level complexity. In Jozef Gruska, editor, *Mathematical Foundations of Computer Science 1977, 6th Symposium, Tatranska Lomnica, Czechoslovakia, September 5-9, 1977, Proceedings*, volume 53 of *Lecture Notes in Computer Science*, pages 162–176. Springer, 1977. doi:10.1007/3-540-08353-7\_135.
- 25 V. N. Vapnik and A. Ja. Červonenkis. The uniform convergence of frequencies of the appearance of events to their probabilities. *Teor. Verojatnost. i Primenen.*, 16:264–279, 1971.



## A

 Total Function Polynomial Hierarchy

► **Definition 21 (TFNP).** A relation  $R(x, y)$  is in **TFNP** if it is polynomial and **total** (for every  $x$  there exists  $y$  such that  $(x, y)$  is in the relation) and there exists a polynomial time Turing machine  $M$  such that  $M(x, y)$  accepts iff  $R(x, y)$  holds.

► **Definition 22 (TFΣ<sub>2</sub>).** A relation  $R(x, y)$  is in **TFΣ<sub>2</sub>** if it is polynomial, total, and there exists a polynomial time Turing machine  $M$  and polynomial  $p(n)$  such that  $R(x, y) \iff \forall z \in \{0, 1\}^{p(|x|)} M(x, y, z)$  accepts.

► **Definition 23 (TFΣ<sub>i</sub>).** A relation  $R(x, y)$  is in **TFΣ<sub>i</sub>** if it is polynomial, total, and there exists a polynomial time Turing machine  $M$  and polynomials  $p(n)_1, \dots, p(n)_{i-1}$  such that  $R(x, y) \iff \forall z_1 \in \{0, 1\}^{p(|x|)_1} \exists z_2 \in \{0, 1\}^{p(|x|)_2} \forall z_3 \in \{0, 1\}^{p(|x|)_3} \dots M(x, y, z_1, z_2, z_3, \dots, z_{i-1})$  accepts.

At this point one may ask, what about a **TFΠ<sub>i</sub>**? Could we define total function complexity classes where the first quantifier is an *exists*? It turns out that such a definition results in a complexity class that is polynomial-time reducible to **TFΣ<sub>i-1</sub>** and vice versa, and hence, does not capture anything new. In this way, the total function hierarchy is different from its decision-problem analogue, where, by the way of oracles,  $\Sigma_{i-1} \neq \Pi_i \neq \Sigma_i$ .

► **Proposition 24.** Let  $R(x, y)$  be a polynomial, total relation such that there exists a polynomial time Turing machine  $M$  and polynomials  $p_1(n), \dots, p_{i-1}(n)$  such that  $R(x, y) \iff \exists z_1 \in \{0, 1\}^{p_1(|x|)} \forall z_2 \in \{0, 1\}^{p_2(|x|)} \exists z_3 \in \{0, 1\}^{p_3(|x|)} \dots M(x, y, z_1, z_2, z_3, \dots, z_{i-1})$  accepts. Every search problem in **TFΣ<sub>i-1</sub>** can be expressed with such a relation, and the search problem for any such relation is polynomial-time reducible to **TFΣ<sub>i-1</sub>**.

**Proof.** The fact that any search problem in **TFΣ<sub>i-1</sub>** can be expressed with such a relation (one that starts with  $\exists$ ) is trivial: the relation is the same, and one can reuse the **TFΣ<sub>i-1</sub>** Turing Machine  $M$ , simply by ignoring  $z_1$ . On the other hand, given a  $R(x, y)$  as above, by totality for every  $x$  there is a  $y$  such that there exists  $z$  satisfying the rest of the condition; hence, the relation  $R(x, (y, z))$  defined by  $R(x, (y, z)) \iff \forall z_2 \in \{0, 1\}^{p(|x|)_2} \exists z_3 \in \{0, 1\}^{p(|x|)_3} \dots M(x, y, z_1, z_2, z_3, \dots)$  is total, and is clearly in **TFΣ<sub>i-1</sub>**. Hence one can solve  $R(x, y)$  with one call to a **TFΣ<sub>i-1</sub>** oracle, obtaining a pair  $(y, z)$  and discarding  $z$ . ◀

In other words, the total function polynomial hierarchy does not have “two symmetric sides” like the classical one, but is a single tower of classes.

Finally, analogously to the decision problem polynomial hierarchy, the total function polynomial hierarchy can be understood through oracles; **TFΣ<sub>i</sub>**  $\subseteq$  **TFNP<sup>Σ<sub>i-1</sub></sup>**, and **TFNP<sup>Σ<sub>i-1</sub></sup>** is polynomial time reducible to **TFΣ<sub>i</sub>**.

► **Theorem 25.** **TFΣ<sub>i</sub>**  $\subseteq$  **TFNP<sup>Σ<sub>i-1</sub></sup>**  $\leq_T^P$  **TFΣ<sub>i</sub>**, where the latter class indicates **TFNP<sup>Σ<sub>i-1</sub></sup>** problems where the verifying Turing Machine has access to a **Σ<sub>i-1</sub>** oracle.

**Proof.** We present the proof for **TFΣ<sub>2</sub>**; the proof for other levels is analogous. The trivial direction is that **TFΣ<sub>2</sub>**  $\subseteq$  **TFNP<sup>Σ<sub>1</sub></sup>**. For a relation  $R(x, y)$  with verifying machine  $M(x, y, z)$  we define a **TFNP<sup>Σ<sub>1</sub></sup>** machine  $M'(x, y)$  which issues a single **Σ<sub>1</sub>** query for whether  $\forall z M(x, y, z)$  accepts, and outputs the answer. For the other direction, let  $R(x, y)$  be a **TFNP<sup>Σ<sub>1</sub></sup>** relation with verifying Turing Machine  $M^{\Sigma_1}(x, y)$  which makes at most  $p(|x|)$  oracle queries in its computation, each of length at most  $p(|x|)$ . Define  $R'(x, (y, \mathbf{a}, \mathbf{z}))$  where  $\mathbf{a} \in \{0, 1\}^{p(|x|)}$ ,  $\mathbf{z} \in \{0, 1\}^{2p(|x|)}$  by whether  $\mathbf{w} \in \{0, 1\}^{2p(|x|)} M'(x, (y, \mathbf{a}, \mathbf{z}, \mathbf{w}))$  accepts, where  $M'$  only accepts if:



#### 44:18 Total Functions in the Polynomial Hierarchy

1.  $M(x, y)$  is an accepting computation given oracle answers  $\mathbf{a}$ ;
2. if the  $i$ -th oracle answer in  $\mathbf{a}$  is a *yes* answer, then the  $i$ -th string in  $\mathbf{z}$  is a satisfying assignment to the  $i$ -th query in the computation  $M(x, y)$  (possibly using only a prefix of  $\mathbf{z}$ );
3. if the  $i$ -th oracle answer in  $\mathbf{a}$  is a *no* answer, then the  $i$ -th string in  $\mathbf{w}$  does not satisfy the  $i$ -th query in the computation  $M(x, y)$ .

Indeed,  $R(x, y) \iff \exists \mathbf{a}, \mathbf{z} R'(x, (y, \mathbf{a}, \mathbf{z}))$ , and the latter is a  $\mathbf{TF}\Sigma_2$  relation; to reduce  $R$  to  $R'$ , compute  $R'$  and discard  $\mathbf{a}, \mathbf{z}$ . ◀

# Relaxing Common Belief for Social Networks

Noah Burrell 

University of Michigan, Ann Arbor, MI, USA  
burrelln@umich.edu

Grant Schoenebeck 

University of Michigan, Ann Arbor, MI, USA  
schoeneb@umich.edu

---

## Abstract

We propose a relaxation of common belief called *factional belief* that is suitable for the analysis of strategic coordination on social networks. We show how this definition can be used to analyze revolt games on general graphs, including by giving an efficient algorithm that characterizes a structural result about the possible equilibria of such games.

This extends prior work on common knowledge and common belief, which has been too restrictive for use in understanding strategic coordination and cooperation in social network settings.

**2012 ACM Subject Classification** Theory of computation → Social networks; Theory of computation → Network games; Theory of computation → Algorithmic game theory

**Keywords and phrases** Social networks, network revolt games, common belief

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.45

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/2009.12936>.

**Funding** The authors gratefully acknowledge the support of the National Science Foundation under Career Award #1452915.

## 1 Introduction

Common knowledge and its analogues (e.g. common belief) are fundamental in the analysis of coordination and cooperation in strategic settings. Informally, common knowledge is the phenomenon that, within a population, everyone knows that a proposition is true, everyone knows that everyone knows that it is true, everyone knows that everyone knows that everyone knows that it is true, and so on, ad infinitum. The existence of common knowledge often shows up as an assumption that underlies some kind of strategic coordination. For example, in the standard game theoretic setting, it is typically assumed that the agents playing a game have common knowledge both of the payoffs of the game and of the rationality of each agent. This assumption undergirds the agents' ability to coordinate on equilibria.

A rigorous investigation into the consequences of this type of assumption can be traced back to Robert Aumann [1], who showed that if two rational agents have the same prior, and their posteriors for an event are common knowledge, then their posteriors must be equal (i.e. it is impossible for such agents to “agree to disagree”). Later work considered whether assumptions about common knowledge, and the mathematical consequences of those assumptions, were realistic for trying to explain economic phenomena. In particular, it seemed unrealistic to expect agents to reason about infinite hierarchies of knowledge. Initial attempts to address this critique involved truncating the infinite hierarchy that is required in our informal definition of common knowledge after some large, but finite, number of levels. However, this line of inquiry resulted in the discovery of “common knowledge paradoxes” that arose from examples like Ariel Rubinstein’s Electronic Mail Game [15]. Such examples demonstrated that in situations where common knowledge was required for coordination, if the infinite hierarchy were truncated to be any finite hierarchy, strategic



© Noah Burrell and Grant Schoenebeck;  
licensed under Creative Commons License CC-BY  
12th Innovations in Theoretical Computer Science Conference (ITCS 2021).  
Editor: James R. Lee; Article No. 45; pp. 45:1–45:20



Leibniz International Proceedings in Informatics  
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

agents behaved unexpectedly and unrealistically. That is, they behaved very differently when a proposition was close to common knowledge, in the sense that many – but only finitely many – hierarchies of knowledge were satisfied, then they did when the proposition was actually common knowledge (and therefore infinitely many hierarchies of knowledge were satisfied).

An important breakthrough came from Dov Monderer and Dov Samet in 1989 [8]. Inspired by an earlier version of Rubinstein’s work [15] and the work of Aumann [1], Monderer and Samet proposed an alternative to truncating the infinite hierarchy of knowledge: relaxing the requirement of “knowledge” at each level in the hierarchy. They introduced the notion of *common belief* – an analogous concept to common knowledge that is defined by replacing “everyone knows” with “everyone believes with probability (at least)  $p$ ” at each place where it occurs in our informal definition of common knowledge. Under this definition, they showed that common belief approximates common knowledge in precisely the way that truncating the infinite hierarchy did not: When common knowledge of some proposition is relaxed to common belief, agents behave in approximately the same way they did when they had common knowledge of the proposition. For example, common belief can be applied to generalize Aumann’s result that was mentioned above: when rational agents have the same prior and their posteriors about an event are commonly believed, these posteriors must be approximately equal.

In the same paper, they also detailed a key insight that allowed them to address the critique that common knowledge might be too unrealistic to have explanatory power in Economics. They showed that common knowledge has an alternative definition (used implicitly in [1]), that is formally equivalent to the more natural definition, but simpler to reason about. This alternative – inspired by the example of public announcements – defines common knowledge in terms of events that are *evident knowledge*: events for which their occurrence implies knowledge of their occurrence within the entire population. Common belief has a similar formally equivalent, alternative definition in terms of events that are *evident belief*: events for which their occurrence implies belief with probability at least  $p$  of their occurrence within the entire population.

Here, it is worth noting that our work most heavily borrows from that of Monderer and Samet and, as such, their paper [8] is highly recommended both as a primer for what follows here and as an introduction to common knowledge and common belief in general.

In a somewhat orthogonal line of research, Michael Suk-Young Chwe studied common knowledge (not common belief) on networks in the context of a revolt game [3, 4]. In his setting, each agent has a threshold that represents the size of the revolt in which she would agree to participate. For example, an agent with a threshold of 3 would need to know that at least 2 other agents would participate in order to herself agree to participate in the revolt. To decide whether this threshold is met, agents learn the thresholds of their neighbors. They use this information (and knowledge of their 2-hop neighborhood in the graph) to reason about which agents have their thresholds satisfied and consequently whether their own threshold is satisfied. Because agents require absolute certainty in this setting, they only consider their neighbors when conducting this reasoning.

The strict requirement that absolute certainty is necessary for an agent to revolt implies that for a group of agents to revolt, it must be common knowledge among them that all of their thresholds are satisfied. Consequently, the problem of finding groups of revolting agents reduces to the problem of finding cliques of a certain size in the network. However, an important innovation of Chwe’s work, in contrast to the settings described above, is that common knowledge in his setting is a local phenomenon occurring within those cliques, not a global phenomenon occurring within the entire population.

Both approaches – that of Monderer and Samet and that of Chwe – are limited in their usefulness in social network settings, because their respective notions are unlikely to apply in many natural graphs that are used to model social networks. For example, an Erdős-Rényi random graph of  $n$  vertices with  $p = \frac{10}{n}$  will almost certainly be sparse, so population-level phenomenon like common belief will not arise. On the other hand, an Erdős-Rényi random graph of  $n$  vertices with  $p = \frac{1}{2}$  would be unlikely to contain large cliques, so the phenomenon of local common knowledge would be severely limited despite the fact that each agent, in seeing about half the graph, should have a lot of information about the entire population and might be expected to be able to coordinate with some large fraction of it.

We propose that Monderer’s and Samet’s concept of common belief – itself a relaxation of common knowledge – can be further relaxed to a notion of common belief among a faction (i.e. a minimal-size subset) of the population, while retaining both its mathematical simplicity (in being defined in terms of events that are evident belief) and its economic explanatory power. We refer to this notion as *factional belief*.

Factional belief is a natural application of the ideas of Monderer and Samet to network settings similar to those of Chwe. It retains from Chwe the idea that common knowledge/belief can occur in only a subset of the population and still motivate their behavior. However, it is not prohibitively strict, such that it would be unlikely to occur endogenously in many natural graphs. Factional belief is not necessarily local – agents can and may need to reason about agents outside of their neighborhood. As such, it does not require cliques.

## 1.1 Our Contributions

- We formally define a notion of factional belief (Section 2), which can be used to analyze revolt games on general graphs (Section 3). Prior notions of common knowledge and common belief were insufficient for this type of analysis.
- We provide an algorithm that characterizes a structural result about the types of equilibria that are possible in instances of the network revolt games described in Section 3 (Section 4) and a natural extension of those games (Section 5).
- We show that, surprisingly, it is sufficient for our algorithm to only have access to the degree sequence of the network; additional details of the network beyond the degree sequence are not relevant.
- In the full version of the paper, we describe algorithms for additional natural extensions of the network revolt game model. Further, we demonstrate the practical utility of our algorithms by applying them to simulated network data to explore how various parameters of networks and of the model relate to the size of revolts that are supported in equilibria of the network revolt game.

## 1.2 Additional Related Work

The work of Stephen Morris is particularly notable when surveying the literature related to common knowledge and common belief. Morris, often following the work of Monderer and Samet, has done much theoretical work relating to common knowledge and common belief [9, 11, 12, 13]. More recently, he has also collaborated with Benjamin Golub to study higher-order reasoning, reminiscent of the infinite hierarchy of reasoning in the initial definitions of common knowledge and common belief, in network settings [6, 7].

Underlying Morris’ work, above, and our work is our assertion that common knowledge, common belief, and factional belief are useful, not just as mathematical concepts, but for understanding real social and economic phenomena. Here, we briefly outline some relevant

work in applying common knowledge and common belief to explaining such phenomena. One clear direction for future research is to try to similarly apply factional belief as an explanatory tool in these and other settings.

Morris has applied common knowledge/belief to settings such as contagion [10] and global games [14].

Chwe, in 2013, revisited his earlier work and expanded his purview to consider how common knowledge is generated in society [5]. He proposed that the importance of rituals in society can be understood from the perspective that rituals create conditions under which common knowledge can be generated. This theory fits nicely with understanding common knowledge through the lens of evident knowledge events. Another potential direction for future research would be to try to mathematically model this ritualistic generation of common knowledge.

Finally, common knowledge, common belief, and factional belief can be used as tools to help understand the formation and transformation of social norms. In particular, Cristina Bicchieri proposed and meticulously advocated for a definition of social norms of which our definition of factional belief is very reminiscent [2].

## 2 Defining Factional Belief

For the following definitions, let  $(\Omega, \Sigma, \text{Pr})$  be a probability space, where  $\Omega$  denotes a set of states,  $\Sigma$  denotes a  $\sigma$ -algebra of events, and  $\text{Pr}$  denotes a probability measure on  $\Sigma$ . Let  $I$  denote a set of agents.

For each  $i \in I$ ,  $\Pi_i$  is a partition of  $\Omega$  into measurable sets with positive probability. It is, therefore, a countable partition. For  $\omega \in \Omega$ , the element of  $\Pi_i$  that contains  $\omega$  is written as  $\Pi_i(\omega)$ .  $\Pi_i$  can be interpreted as the information available to agent  $i$ . That is,  $\Pi_i(\omega)$  is the set of states that are indistinguishable to  $i$  when  $i$  observes  $\omega$ . Let  $B_i^p(E)$  denote the event that agent  $i$  believes in event  $E$  with probability at least  $p$ . Formally, we write  $B_i^p(E) = \{\omega : \text{Pr}[E|\Pi_i(\omega)] \geq p\}$ . Lastly, for events  $E$  and  $F$ , we use the notation  $E \subseteq F$  to denote that, whenever  $E$  occurs,  $F$  occurs.<sup>1</sup>

The following examples are helpful to illustrate this notation. When rolling a fair die, with equally probable outcomes in the set  $\{1, 2, 3, 4, 5, 6\}$  we have  $\{2, 4\} \subseteq \{\text{Outcome is even}\}$ . Similarly, when the die has been tossed, but the outcome has not been revealed, we have the event  $B_i^{\frac{1}{2}}(\{\text{Outcome is even}\})$  for any agent  $i$ .

This notation is borrowed from Monderer and Samet [8], and the rest of the terms and claims in this section are defined and stated, respectively, to be analogous to those from their paper.

► **Definition 1** (Evident  $(p, \mu)$ -belief). *An event  $E$  is an evident  $(p, \mu)$ -belief if there exists (at least) a  $\mu$  fraction of agents such that whenever  $E$  occurs, those agents assign a probability of at least  $p$  to its occurrence. That is:*

$$\exists J \subseteq I \text{ with } |J| \geq \mu|I| \text{ such that for each } j \in J, E \subseteq B_j^p(E).$$

Following, Monderer and Samet, we first define our notion of factional belief in terms of events that are evident  $(p, \mu)$ -belief. To maintain consistency with their work, we refer to this notion of factional belief as common  $(p, \mu)$ -belief.

<sup>1</sup> Note that, unlike with knowing, it is possible for an agent to believe something that is not true. In particular,  $B_i^p(E)$  need not be a subset of  $E$ .

► **Definition 2** (Common  $(p, \mu)$ -belief). *An event  $F$  is common  $(p, \mu)$ -belief at  $\omega \in \Omega$  if there exists an evident  $(p, \mu)$ -belief event  $E$  such that  $\omega \in E$  and*

$$\exists J \subseteq I \text{ with } |J| \geq \mu|I| \text{ such that for each } j \in J, E \subseteq B_j^p(F).$$

That is,  $F$  is common  $(p, \mu)$ -belief whenever there is an event ( $E$ ) that is an evident  $(p, \mu)$ -belief whose occurrence implies the existence of (at least) a  $\mu$  fraction of agents that believe with probability at least  $p$  in  $F$ . Note that any event  $E$  that is an evident  $(p, \mu)$ -belief is trivially also common  $(p, \mu)$ -belief (with  $F = E$ ).

Now, an important property shared by common knowledge and common belief is the formal equivalence of their definitions in terms of evident events and their intuitive definitions as infinite hierarchies. Common  $(p, \mu)$ -belief retains this property. In order to state this result formally in Proposition 4, we need to formally define the infinite hierarchy, which is done below in Definition 3.

Informally, each level ( $n \geq 1$ ) in this hierarchy, refers to the event that there exists (at least) a  $\mu$  fraction of agents who believe with probability (at least)  $p$  in the previous level of the hierarchy. The initial level ( $n = 0$ ) is simply the relevant event  $F$ . So, written out entirely, the full informal definition would be that an event  $F$  is common  $(p, \mu)$ -belief if there exists a  $\mu$  fraction of agents who believe  $F$  with probability  $p$ , there exists a  $\mu$  fraction of agents who believe with probability  $p$  that there exists a  $\mu$  fraction of agents who believe  $F$  with probability  $p$ , and so on, ad infinitum.

► **Definition 3.** *For every event  $F$  and every  $0 \leq p \leq 1$  let*

$$E^{p,\mu}(F) = \bigcap_{n \geq 1} F_\mu^n,$$

where  $F_\mu^0 = F$  and  $F_\mu^n$  is the event “ $\exists J \subseteq I$  with  $|J| \geq \mu|I|$  such that  $\forall j \in J, B_j^p(F_\mu^{n-1})$ ”.

► **Proposition 4.** *For every event  $F$ , every  $0 \leq p \leq 1$ , and every  $0 \leq \mu \leq 1$ :*

1.  $E^{p,\mu}(F)$  is an evident  $(p, \mu)$ -belief and  $\exists J \subseteq I$  with  $|J| \geq \mu|I|$  such that  $\forall j \in J, E^{p,\mu}(F) \subseteq B_j^p(F)$ .
2.  $F$  is common  $(p, \mu)$ -belief at  $\omega$  if and only if  $\omega \in E^{p,\mu}(F)$ .

The proof of this proposition is essentially the same as the proof of the analogous proposition (Proposition 2) in Monderer’s and Samet’s paper [8]. However, the details of the proof are not particularly relevant or instructive with regard to our contributions in this work, so we omit them here and consign them to an appendix of the full version of the paper. The important takeaway from this proposition, is, as noted above, the formal equivalence between the definition of common  $(p, \mu)$ -belief in terms of evident  $(p, \mu)$ -belief (Definition 2) and the hierarchical definition (Definition 3). As with the analogous definitions for common knowledge and common belief, the latter definition is more intuitive and perhaps more natural, but the former definition is more mathematically convenient and is what we will reference in what follows. The former definition is also a notion that better corresponds to how agents might be expected to reason about this type of belief in reality, since it is unrealistic to suppose that they consider infinite hierarchies of beliefs.

### 3 Model

Let  $G = (V, E)$  be a graph and  $I$  be a set of  $n$  agents, such that each vertex  $v_i \in V$  corresponds to an agent  $i \in I$ . We will think of  $G$  as representing a social network of strategic agents who are participating in a revolt game. The graph  $G$  is common knowledge among the agents.

Nature draws the state of the world  $s \in S$  according to a distribution  $D_S$  and selects a type  $t_i \in T = \{\alpha, \nu\} \cup X$  for each agent  $i$ , where  $X = \cup_j \{\chi_j\}$  is non-empty (and finite). The types are selected independently at random according to a distribution  $D_T^s$  associated with state  $s$ .  $D_S$  and  $D_T^s$  for each  $s \in S$  are common knowledge among the agents.

Each agent  $i$  will observe the type of each agent  $k \in I$  such that  $(v_i, v_k) \in E$  (this set of agents constitutes the set of *neighbors* of  $i$ ). The information resulting from this observation – an agent’s type and the types of all of her neighbors – defines that agent’s *context*. When agent  $i$  has the context  $c$ , we write it as  $c(i) = c$ . We use  $C$  to denote the set of all contexts that are possible in  $G$ .

*Ex ante*, or, before selection of the state, the assignment of types, and the observation of contexts, agents choose a pure *strategy*  $\sigma : C \rightarrow \{R, Y\}$ , where  $\{R, Y\}$  is the set of *actions*.<sup>2</sup> A *strategy profile*  $(\sigma_1, \sigma_2, \dots, \sigma_n)$  is collection of strategies for each agent. Let  $\sigma_{-i} = (\sigma_1, \sigma_2, \dots, \sigma_{i-1}, \sigma_{i+1}, \dots, \sigma_n)$  denote the strategies of all the agents except for  $i$ . Similarly, *ex post*, or, after the selection of types and observation of contexts, an *action profile*  $(a_1, a_2, \dots, a_n)$  where  $a_i = \sigma_i(c(i))$  is a collection of actions for each agent.

Let  $\mathcal{R}(a_1, a_2, \dots, a_n) = \{i \in [n] : a_i = R\}$  denote the set of agents who play the action  $R$  (revolt) given their context and strategy. Agent  $i$  with type  $t_i = t$  receives a payoff according to the function  $f_i^t : \{R, Y\}^n \rightarrow [0, 1]$ :

$$f_i^\alpha(a_1, a_2, \dots, a_n) = \begin{cases} 1, & \text{if } a_i = R, \\ 0, & \text{otherwise,} \end{cases}$$

$$f_i^\nu(a_1, a_2, \dots, a_n) = \begin{cases} 1, & \text{if } a_i = Y, \\ 0, & \text{otherwise,} \end{cases}$$

and

$$f_i^{\chi_j}(a_1, a_2, \dots, a_n) = \begin{cases} 1 - p_j, & \text{if } |\mathcal{R}(a_1, a_2, \dots, a_n)| \geq \mu_j \cdot n \text{ and } a_i = R, \\ p_j, & \text{if } |\mathcal{R}(a_1, a_2, \dots, a_n)| < \mu_j \cdot n \text{ and } a_i = Y, \\ 0, & \text{otherwise.} \end{cases}$$

where  $p_j, \mu_j \in [0, 1]$  for each  $j$  and are common knowledge among the agents.

We call  $\sigma_i$  a *best-response* to  $\sigma_{-i}$  when  $\sigma_i$  maximizes agent  $i$ ’s *ex ante* expected payoff given  $\sigma_{-i}$ , which, by linearity of expectation, is equivalent to maximizing her expected payoff for each  $c \in C$  (using the beliefs she would have about the contexts of the other agents after observing  $c$ ).

We say that a strategy profile  $(\sigma_1, \sigma_2, \dots, \sigma_n)$  is an *equilibrium* when each strategy  $\sigma_i$  is a best-response to  $\sigma_{-i}$ .

Intuitively, the game is defined so that there is a natural strategy for each agent based on type:

- Agents of type  $\alpha$  should always revolt.
- Agents of type  $\nu$  should never revolt.
- Agents of type  $\chi_j$  should conditionally revolt:
  - Each type  $\chi_j$  is characterized by a pair of thresholds  $(p_j, \mu_j)$  that indicate an agent of type  $\chi_j$  should revolt when she believes that  $\Pr[|\mathcal{R}(a_1, a_2, \dots, a_n)| \geq \mu_j] \geq p_j$ .

An important thing to note about this game is that when there are agents of type  $\nu$ , there is not an equilibrium where each agent revolts regardless of her beliefs. Similarly, when there are agents of type  $\alpha$ , there is not an equilibrium where each agent chooses not to revolt regardless of her beliefs.

<sup>2</sup> “Revolt” and “Yield,” respectively.



Even without these, though, there are still potentially multiple equilibria corresponding to revolts of different sizes, and our work does not address equilibrium selection. Consequently, in what follows, we will write that a revolt (of a particular size) is *supported* in equilibrium instead of writing that a revolt will occur. Similarly, we write that agents are *secure enough to revolt* when they sufficiently believe their thresholds are met instead of writing that agents will revolt.

Lastly, in this work we primarily consider the *largest* revolts that are supported in some equilibrium. Such revolts are supported by *symmetric* equilibria, in which each agent adopts the same strategy – namely, the strategy detailed in the bullet points above.

### 3.1 Motivating Example

To see this model in practice and get a feeling for how agents need to reason about the concepts that we have introduced, we will work through a modest example.

Suppose that  $G$  is a grid of  $n$  vertices embedded on a torus, so that each the vertex associated with each agent is adjacent to the vertices representing exactly four other agents (and there is no boundary). Thus, a context in this graph consists of the central agent and her type plus the types of each of her neighbors in  $G$ . In this example, we will have two types of agents: (1) Agents of type  $\chi$ , who want to revolt conditionally; they feel secure enough to revolt when the threshold pair  $(p = \frac{2}{5}, \mu = \frac{1}{2})$  is satisfied. (2) Agents of type  $\nu$ , who never want to revolt. We will also have two equally-likely states: an anti-government state  $A$ , where agents are of type  $\chi$  with probability  $\frac{4}{5}$  and a pro-government state  $B$  where agents are of type  $\nu$  with probability  $\frac{4}{5}$ .

We also define the notion of a *candidate agent*. In this setting, for reasons that will become clear in the proof of the following proposition, we refer to agents of type  $\chi$  with two or more neighbors of type  $\chi$  as *candidate agents*.

► **Proposition 5.** *In this model, when  $G$  is sufficiently large, the event that at least  $\frac{1}{2}$  of the agents are candidate agents is an evident  $(\frac{2}{5}, \frac{1}{2})$ -belief. When it occurs, it is common  $(\frac{2}{5}, \frac{1}{2})$ -belief that supports a revolt of size  $\frac{1}{2}$ .*

**Proof.** First, we will compute  $\Pr[\text{State} = A|c]$  for each context  $c$  and use the computed values to demonstrate that our definition of candidate agents is the correct one for this setting; candidates are likely to feel secure enough to revolt, based on their  $p$ -thresholds.

Each agent has 5 independent samples from the probability distribution defined by the state, with which to calculate the probability of each state, given her information. The first step in this is to compute the likelihood of her context given the state, which can be calculated by evaluating the probability mass function of a binomial distribution with the appropriate parameters. Then, the probability for each state given a certain neighborhood can be calculated using Bayes' rule. The results of these calculations are shown in Table 1.

Now, agents of type  $\chi$  that only have at most one other neighbor of type  $\chi$ , believe that the state is  $A$  with probability at most  $\frac{1}{5}$ , so they will not feel secure enough to revolt; a revolt of size  $\mu = \frac{1}{2}$  is very unlikely to be supported when the state is  $B$ , and they do not sufficiently believe that the state is  $A$ .

As a result, it is exactly agents of type  $\chi$  with contexts such that they have two or more neighbors of type  $\chi$ , which we defined earlier as candidate agents, in which we are interested (cases with  $k \geq 3$  in Table 1b). In particular, we are interested in the probability that a majority of agents are candidates.

▷ **Claim.** The probability that at least half the agents are candidates, given that the state is  $A$  is at least  $\frac{1739}{3125}$ .

Proof of Claim. For this, we can try to count all the non-candidate agents and see how likely they are to outnumber the candidates.

Let  $X$  count the number of non-candidate agents. Specifically, let  $X = \sum_{i \in [n]} X_i$ , where  $X_i$  is an indicator variable that is 1 if and only if agent  $i$  is not a candidate. For each  $X_i$ , therefore, the expectation of  $X_i$  is equal to the probability that  $i$  is not a candidate, which can happen in three ways: (1)  $i$  is of type  $\nu$ , (2)  $i$  is of type  $\chi$  and has no neighbors of type  $\chi$ , or (3)  $i$  is of type  $\chi$  and has one neighbor of type  $\chi$ . The probability of (1) is trivially  $\frac{1}{5}$ . Conditioned on  $i$  being of type  $\chi$ , the respective probabilities of the remaining possibilities are (2)  $\frac{1}{625}$  and (3)  $\frac{16}{625}$ . To obtain unconditional probabilities for (2) and (3), we multiply each conditional probability by  $\frac{4}{5}$ . Consequently, since the events (1), (2), and (3) are disjoint, we have the following:

$$\mathbb{E}[X_i] = \frac{1}{5} + \frac{4}{5} \left( \frac{1}{625} \right) + \frac{4}{5} \left( \frac{16}{625} \right) = \frac{693}{3125}.$$

Therefore, by linearity of expectation,  $\mathbb{E}[X] = \frac{693n}{3125}$ .

Now, we need to upper bound the probability that the number of non-candidate agents is the majority. Ideally for this, we would like to use something tight, like a typical Chernoff-Hoeffding bound. Unfortunately, here, the  $X_i$ 's are not independent; the contexts of agents that share neighbors are correlated, which complicates the calculation.

For our more rigorous analysis later, we will insist on tighter probability bounds. But for the sake of simplicity, in this example, Markov's inequality is sufficient:

$$\Pr \left[ X \geq \frac{n}{2} \right] \leq \frac{\frac{693n}{3125}}{\frac{n}{2}} = \frac{1386}{3125} \approx 0.44.$$

Consequently, the probability that at least half the agents are candidates, given that the state is  $A$ , is at least  $(1 - \frac{1386}{3125}) = \frac{1739}{3125}$ .  $\triangleleft$

Candidate agents can perform this exact same calculation and, further, they believe that the state is  $A$  with probability at least  $\frac{4}{5}$ . Thus, when the graph is large enough that the context of any individual agent is inconsequential in their reasoning about the total fraction of candidate agents, the probability that they assign to at least half of the agents being candidate agents is at least  $\frac{4}{5} \cdot (\frac{1739}{3125}) \geq \frac{2}{5}$ .

That is, in sufficiently large graphs, candidate agents believe with probability at least  $\frac{2}{5}$  that at least  $\frac{1}{2}$  of the agents are candidate agents. As a result, the event that at least  $\frac{1}{2}$  of the agents are candidate agents is an evident  $(\frac{2}{5}, \frac{1}{2})$ -belief. Consequently, when it occurs, by definition, it is common  $(\frac{2}{5}, \frac{1}{2})$ -belief. In this event, a revolt of size  $\frac{1}{2}$  is supported, since at least half of the agents have their thresholds satisfied, and consequently feel secure enough to revolt.  $\blacktriangleleft$

## 4 Applying Factional Belief: A Computational Perspective

The network revolt game model described in Section 3 gives us a useful setting in which to apply our definitions from Section 2 and demonstrate how they can provide insight into strategic coordination. In this section, we explore the interaction between factional belief and strategic coordination from a computational perspective, by trying to answer an elementary question: When can we efficiently determine if strategic coordination (i.e. in our model, a revolt) is supported under given conditions?

■ **Table 1** Results of the probabilistic calculations for the motivating example.

(a) The likelihood, in each state, of having a context with  $k$  agents of type  $\chi$ .      (b) The likelihood of state  $A$ , given a context with  $x$  agents of type  $\chi$ .

	$\Pr[c \text{State} = A]$	$\Pr[c \text{State} = B]$
$k = 0$	$\frac{1}{3125}$	$\frac{1024}{3125}$
$k = 1$	$\frac{4}{625}$	$\frac{256}{625}$
$k = 2$	$\frac{32}{625}$	$\frac{128}{625}$
$k = 3$	$\frac{128}{625}$	$\frac{32}{625}$
$k = 4$	$\frac{256}{625}$	$\frac{4}{625}$
$k = 5$	$\frac{1024}{3125}$	$\frac{1}{3125}$

	$\Pr[\text{State} = A k = x]$
$x = 0$	$\frac{1}{1025}$
$x = 1$	$\frac{1}{65}$
$x = 2$	$\frac{1}{5}$
$x = 3$	$\frac{4}{5}$
$x = 4$	$\frac{64}{65}$
$x = 5$	$\frac{1024}{1025}$

In pursuing an answer to this question, we seek to apply the intuition that we have constructed in our motivating example to a more general setting. Toward that end, a relatively modest generalization of the model used in the example – our Fundamental Case, below – is rich enough to provide an interesting, non-trivial answer to our question and to provide robust intuition that guides us through the various extensions of the model that we discuss in Section 5.

#### 4.1 Fundamental Case: Low-Degree Graphs, Two States, and Three Types

Recall that our model requires a description of possible agent types and possible states (which specify probability distributions over those types). For our initial case, there are three agent types and two possible states.

Our focus will be on agents who want to revolt conditionally – agents of type  $\chi$  – who have the threshold pair  $(p, \mu)$  for an arbitrary  $0 \leq p \leq 1$  and  $0 \leq \mu \leq 1$ . In some sense, these are the truly “strategic” agents; they need to coordinate with other agents in order to feel secure enough to revolt. There are also agents who always behave in a prescribed manner, regardless of their contexts or the state: pro-government agents of type  $\nu$ , who will never revolt, and anti-government agents of type  $\alpha$ , who will always revolt.

The possible states are  $A$ , an anti-government state, and  $B$ , a pro-government state.  $D_S$ , the distribution over the states,  $D_T^A$  and  $D_T^B$ , the distributions over the types in each state, and  $p$  and  $\mu$ , the threshold values for agents of type  $\chi$ , are commonly known to the agents under the prior  $P$ . Intuitively, the labels assigned to the states as being anti- and pro-government correspond to an implication that as the number of agents grows, the size of the largest revolt supported in state  $A$  should be larger than in state  $B$ . In what follows, we assume that the given labels are assigned correctly, but in practice the correct labels can be determined by running an algorithm that we present later with each possible labeling and comparing the results: When the states are correctly labeled, Algorithm 1 will return  $X_A$  and  $X_B$  such that  $X_A \geq X_B$ .

In this more concrete setting, we are able to pose a straightforward question: Given values  $\mu^*$  and  $q^*$ , is a revolt of size (at least)  $\mu^*$  supported with probability at least  $q^*$ ? We refer to this problem as REVOLT.

► **Definition 6 (REVOLT).**

**Given:**  $(G, P, \mu^*, q^*)$ , where  $G$  is a graph with  $n$  vertices and  $P = (p, \mu, D_T^A, D_T^B, D_S)$  is the common prior.

**Question:** Is a revolt of size at least  $\mu^*$  supported with probability at least  $q^*$ ?

Note that there is a nuance regarding the timing of the network revolt game model described in Section 3. REVOLT considers the likelihood of revolt of a certain size being supported in a given network *ex ante* – i.e. before the selection of the state and the assignment of types to agents.

Although the question posed by REVOLT is straightforward to state, it is not straightforward to solve efficiently. In fact, we have the following hardness result, which we prove in an appendix of the full version of the paper:

► **Proposition 7.** *REVOLT is NP-hard.*

Consequently, we will instead consider a relaxed version of the problem that we will be able to solve efficiently. In order to define this new problem, we first require  $p$  and  $\mu$  to be strictly between 0 and 1. We will also need two error terms, which will be constants given as input:  $\epsilon, \delta > 0$ . Lastly, we introduce two additional priors, as follows (recall  $P = (p, \mu, D_T^A, D_T^B, D_S)$  is the common prior given as input to both problems):

$$P^- = (p - \delta, \mu - \epsilon, D_T^A, D_T^B, D_S), \quad P^+ = (p + \delta, \mu + \epsilon, D_T^A, D_T^B, D_S).$$

Now, we are ready to define our new problem, PROMISE REVOLT.

► **Definition 8** (PROMISE REVOLT).

**Given:**  $(G, P, \mu^*, \epsilon, \delta)$ , where  $G$  is a graph with  $n$  vertices,  $P = (p, \mu, D_T^A, D_T^B, D_S)$  is the common prior, and  $\epsilon, \delta > 0$  are constants.

**Output:** When exactly one of the following cases is true, output the corresponding symbol:  
 $\Omega$ : A revolt of size  $\mu^* - \epsilon$  is supported with probability  $q^* \geq 1 - \delta$  under the prior  $P^-$ .

$A$ : A revolt of size  $\mu^*$  is supported with probability  $q^* \in [\Pr[\text{State} = A] - \delta, \Pr[\text{State} = A] + \delta]$  under the prior  $P$ .

$\emptyset$ : A revolt of size  $\mu^* + \epsilon$  is supported with probability  $q^* \leq \delta$  under the prior  $P^+$ .

PROMISE REVOLT is named so as to emphasize that this it is a promise problem, in the typical sense. The “promise” is that the given instance is such that exactly one of the cases from the definition of the problem is true. Given that promise, the three cases are mutually exclusive and any solution is required to always output the correct answer. It is exactly this promise to exclude difficult inputs that makes PROMISE REVOLT easier to solve than REVOLT.

Still, in providing an algorithm to solve PROMISE REVOLT, we will see that the problem has a structure that allows us to closely approximate REVOLT by solving PROMISE REVOLT (for large graphs). This structure is discussed in more detail at the end of this subsection (4.1), particularly with respect to Figure 1, which helps to illustrate this intuition.

Lastly, note that we assume all inputs to REVOLT and PROMISE REVOLT are given as rational numbers. This leads us to state our first theorem:

► **Theorem 9.** *Given  $\epsilon > 0$ , the prior  $P$ , and  $\mu^* \in [0, 1]$ , there exists  $\delta(n) \in \frac{1}{\exp(\Omega_{\epsilon, P}(\sqrt[3]{n}))}$  such that for any graph  $G$  with  $n$  vertices where the largest degree of any vertex is  $O_{\epsilon, P}(\sqrt[3]{n})$ , Algorithm 3 can be used to solve PROMISE REVOLT( $G, P, \mu^*, \epsilon, \delta = \delta(n)$ ) in polynomial time.*

Before we get to the proof of Theorem 9, we briefly discuss our assumption that each agent has a degree that is  $O(\sqrt[3]{n})$ . Primarily, this assumption serves to simplify our analysis. For simplicity, it is convenient to make some distinction between “low-degree” agents and “high-degree” agents to highlight the fact that a prototypical high-degree agent would have a lot more information about the state than a prototypical low-degree agent. However, any

■ **Algorithm 1** Finding the expected size of largest revolt supported in each state.

---

**Input:** A graph  $G = (V, E)$  of  $n$  vertices, the prior  $P = (p, \mu, D_T^A, D_T^B, D_S)$ .  
**Output:**  $X_A$  and  $X_B$ , the expected size of the largest revolt supported in states  $A$  and  $B$ , respectively.

Let  $e_s(\tau)$  denote the expected fraction of type  $\tau$  agents in state  $s$ .  
Define the set of candidate states  $S_C = \{s \in \{A, B\} : e_s(\chi \cup \alpha) \geq \mu\}$ .

**if**  $S_C = \emptyset$  **then**  
     $X_s = e_s(\alpha) \forall s \in \{A, B\}$ .  
**end**

**if**  $S_C = \{A, B\}$  **then**  
     $X_s = e_s(\chi \cup \alpha) \forall s \in \{A, B\}$ .  
**end**

**if**  $S_C = \{A\}$  **then**  
    Let  $C(\chi)$  be the set of contexts centered around agents of type  $\chi$ .  
    Define the set of candidate contexts  $C_C = \{c \in C(\chi) : \Pr[\text{State} = A|c] \geq p\}$ .  
    (Below, we slightly abuse notation, treating  $C_C$  as if it were a type when writing  $e_s(C_C)$ .)  
    **if**  $e_A(C_C \cup \alpha) \geq \mu$  **then**  
         $X_s = e_s(C_C \cup \alpha) \forall s \in \{A, B\}$ .  
    **end**  
    **else**  
         $X_s = e_s(\alpha) \forall s \in \{A, B\}$ .  
    **end**  
**end**

**return**  $X_A, X_B$ .

---

■ **Algorithm 2** Comparing the expected size of the largest revolt supported in each state.

---

**Input:**  $X_A, X_B$  (output from Algorithm 1) and  $\mu^*$ .  
**Output:**  $\Omega$ ,  $\mathbf{A}$ , or  $\emptyset$ .

**if**  $X_A \geq \mu^*$  **and**  $X_B \geq \mu^*$  **then**  
    **return**  $\Omega$   
**end**

**if**  $X_A \geq \mu^*$  **and**  $X_B < \mu^*$  **then**  
    **return**  $\mathbf{A}$   
**end**

**if**  $X_A < \mu^*$  **and**  $X_B < \mu^*$  **then**  
    **return**  $\emptyset$   
**end**

---

---

**Algorithm 3** Solving PROMISE REVOLT.

---

**Input:**  $(G, P, \mu^*, \epsilon, \delta)$ , where  $G$  is a graph of  $n$  vertices,  $P = (p, \mu, D_T^A, D_T^B, D_S)$  is the common prior, and  $\epsilon, \delta > 0$  are constants

**Output:**  $\Omega$ ,  $\mathbf{A}$ ,  $\emptyset$ , or **Null**.

$X_{A_1}, X_{B_1} \leftarrow \text{Algorithm 1}(G, P = (p', \mu', D_T^A, D_T^B, D_S))$ , where  $p' = p + \frac{\delta}{3}$ ,  
 $\mu' = \mu + \frac{\epsilon}{3}$ .

$S_1 \leftarrow \text{Algorithm 2}(X_{A_1}, X_{B_1}, \mu^*)$ .

$X_{A_2}, X_{B_2} \leftarrow \text{Algorithm 1}(G, P = (p', \mu', D_T^A, D_T^B, D_S))$ , where  $p' = p - \frac{\delta}{3}$ ,  
 $\mu' = \mu - \frac{\epsilon}{3}$ .

$S_2 \leftarrow \text{Algorithm 2}(X_{A_2}, X_{B_2}, \mu^*)$ .

**if**  $S_1 = S_2$  **then**  
     **return**  $S_1$

**end**

**else**

**return** **Null**

**end**

---

particular choice of cutoff to separate high- and low-degree agents is somewhat arbitrary. Here, we choose  $O(\sqrt[3]{n})$ , which is mathematically convenient for defining an upper bound on our error function  $\delta(n)$  in Theorem 9.

Given this cutoff, we focus first on the case where there are only low-degree agents, which is sufficient to provide the guiding intuition that we will follow in the next section, when we discuss broadening the setting in various ways. One such extension will involve allowing vertices of arbitrary degree.

We now proceed by making several arguments which form the building blocks of the proof of Theorem 9 that follows.

► **Lemma 10.** *Algorithms 1 and 2 terminate in polynomial time with respect to the number of agents.*

**Proof.** The key for this property of Algorithm 1 is that contexts are identity-agnostic, so the number of contexts is polynomial in  $n$  when the number of types is a constant. Therefore, we are able to enumerate all of the possible contexts in polynomial time. For each of these contexts  $c$ , we can compute the relevant probabilities –  $\Pr[\text{State} = s|c]$  and  $\Pr[c|\text{State} = s]$  for both states  $s$  – using Bayes’ rule. The rest of the steps in the algorithm are linear in the number of contexts, and therefore also computable in polynomial time. Algorithm 2 is trivially computable in constant-time. ◀

Having shown that they are polynomial-time computable, we now demonstrate the correctness of Algorithms 1 and 2 under idealized conditions.

▷ **Claim.** When agents believe with probability 1 that the actual size of the largest supported revolt in each state will exactly equal its expected size, then Algorithm 1 correctly computes the expected size of the largest revolt in each state. When, further, it is true that the actual size of the largest supported revolt in each state will exactly equal its expected size, then Algorithm 2 identifies the set of states in which revolt of size  $\mu^*$  is supported with no error.

**Proof of Claim.** Algorithm 1 first defines the set of candidate states – these are the states in which it is possible, but not necessarily the case, that type- $\chi$  agents will feel secure enough to revolt, because there are at least  $\mu$   $\alpha$ - and  $\chi$ -type agents. If there are no such states, then

only agents of type  $\alpha$  will revolt. If both states are candidates, then all  $\alpha$ - and  $\chi$ -type agents will feel secure enough to revolt: Agents of type  $\alpha$  always revolt and agents of type- $\chi$  have their  $\mu$  threshold met in both states, by the definition of a candidate state. As a result, their  $p$  threshold is also necessarily met, because the probability of the state being either  $A$  or  $B$  is  $1 \geq p$ . The most interesting case is when only  $A$  is a candidate state. In this case, only type  $\chi$  agents who  $p$ -believe that the state is  $A$  will have their  $p$ -threshold met. So, similarly to our example from Section 3.1, we call those agents candidate agents (and refer to their contexts as candidate contexts.) If the number of candidate agents and  $\alpha$ -type agents, given that the state is  $A$ , is at least  $\mu$ , then all of those agents feel secure enough to revolt. This is true in any state, because even when the state is  $B$ , candidate agents, by definition,  $p$ -believe that the state is  $A$ .

Algorithm 2 simply compares the size of the expected revolt in each state to  $\mu^*$  to decide in which states, if any, a revolt of expected size at least  $\mu^*$  is supported. If the actual size of the revolt in any state is exactly equal to its expectation, as we assume, then Algorithm 2 introduces no error.  $\triangleleft$

Now, the assumption that the actual supported revolt exactly equals the expected size of the supported revolt is, of course, too strict. However, we will be able to show that the actual size of the supported revolt concentrates around its expectation, and consequently, the errors in our algorithms decrease quickly as  $n$  grows.

Recall that computing the expected size of a supported revolt involves counting the number of some subset of three kinds of agents:  $\alpha$ -type agents,  $\chi$ -type agents, and candidate agents. For  $\alpha$ -type and  $\chi$ -type agents, it is simple to show that the number of such agents concentrates around its expectation because agent types are assigned independently at random.

► **Lemma 11.** *Given  $\epsilon > 0$ , the probability that, in a given state, the number of agents of type  $\chi$  and of type  $\alpha$  differ from the expected number of agents of type  $\chi$  and of type  $\alpha$  by a multiplicative factor of  $\epsilon$  is at most  $\delta = \delta(n)$  for some  $\delta(n) \in \frac{1}{\exp(\Omega_{\epsilon,P}(\sqrt[3]{n}))}$ .*

**Proof.** Because the agent types are drawn independently at random, we can use a standard Chernoff-Hoeffding bound on the expected number of each type of agents. Consequently, the difference between the actual and expected number of each type of agent decays exponentially with  $n$ , and therefore we can choose  $\delta(n) \in \frac{1}{\exp(\Omega_{\epsilon,P}(\sqrt[3]{n}))}$  so that the difference is trivially smaller.  $\blacktriangleleft$

Counting the number of candidate agents, as foreshadowed in the Motivating Example of Section 3.1, is somewhat more complicated. Here, we need to consider the contexts of agents – not just their type – and agents' contexts are correlated with the contexts of their neighbors and their neighbors' neighbors in  $G$ . We will show, though, that as the graph grows, the effect of this correlation is small. In fact, we will still be able to prove an exponentially-decreasing bound on the difference between the actual and expected number of candidate agents.

► **Lemma 12.** *Given  $\epsilon > 0$ , the probability that, in a given state, the number of candidate agents is less than the expected number of candidate agents by a multiplicative factor of  $\epsilon$  is at most  $\delta = \delta(n)$  for some  $\delta(n) \in \frac{1}{\exp(\Omega_{\epsilon,P}(\sqrt[3]{n}))}$ .*

**Proof.** Let  $X_C$  be a random variable that denotes the number of candidate agents. We can write  $X_C = \sum_{i=1}^n X_{i,C}$  where  $X_{i,C}$  is an indicator variable that indicates whether or not agent  $i$  is a candidate.



As we have noted above, an agent's status as a candidate is dependent on her context, and as a result, the variables  $X_{i,C}$  are not independent. However – given our assumption that the maximum degree of any vertex in  $G$  is  $O(\sqrt[3]{n})$  – their dependence is constrained enough that we are able to apply a useful exponential bound involving the *fractional chromatic number*  $\chi^*(\Gamma)$  of the constraint graph  $\Gamma$  of the random variables  $X_{i,C}$ .

(See the appendix section of the full version of the paper for additional details).

For our purposes, it is sufficient to use a trivial bound on the fractional chromatic number of any graph. The fractional chromatic number of a graph is at most the *chromatic number* of the graph, which is at most the maximum degree of any vertex plus one. So in  $\Gamma$ , where edges correspond to dependencies between pairs of random variables  $(X_{i,C}, X_{j,C})$ , the maximum degree of any vertex – and therefore the fractional chromatic number  $\chi^*(\Gamma)$  – is  $O_{\epsilon,P}(n^{\frac{2}{3}})$ . Applying the bound, then, gives

$$\Pr[|X_C - \mathbb{E}[X_C]| \geq \epsilon n] \leq 2 \exp\left(\frac{-2\epsilon^2 n}{\chi^*(\Gamma)}\right) \in \frac{1}{\exp(\Omega_{\epsilon,P}(\sqrt[3]{n}))}. \quad \blacktriangleleft$$

► **Lemma 13.**

1. Suppose Algorithm 1 is run with the given inputs, but with modified  $\mu' = \mu + \frac{\epsilon}{3}$  and  $p' = p + \frac{\delta}{3}$ , and Algorithm 2 is run with the resulting  $X_A$  and  $X_B$  and the given  $\mu^*$ . If the result is  $\Omega$ , then a revolt of size  $\mu^* - \epsilon$  is supported with probability at least  $1 - \delta$  under the prior  $P$ . If the result is  $\mathbf{A}$ , then a revolt of size  $\mu^* - \epsilon$  is supported with probability at least  $\Pr[\text{State} = A] - \delta$  under the prior  $P$ .
2. On the other hand, suppose Algorithm 1 is run with the given inputs, with modified  $\mu' = \mu - \frac{\epsilon}{3}$  and  $p' = p - \frac{\delta}{3}$ , and Algorithm 2 is run with the resulting  $X_A$  and  $X_B$  and the given  $\mu^*$ . If the result is  $\emptyset$ , then a revolt of size  $\mu^* + \epsilon$  is supported with probability at most  $\delta$  under the prior  $P$ . If the result is  $\mathbf{A}$ , then a revolt of size  $\mu^* + \epsilon$  is supported with probability at most  $\Pr[\text{State} = A] + \delta$  under the prior  $P$ .

**Proof.** This proof decomposes into two analogous arguments about (1) and (2), which themselves each contain two analogous arguments. We include only the first one in detail here, below, and claim the rest via analogy.

▷ **Claim.** Suppose that, in instance (1), the result is  $\Omega$ . Then, the probability that a revolt of size  $\mu^* - \epsilon$  is supported in both states is at least  $1 - \frac{1}{\exp(\Omega(\sqrt[3]{n}))}$ .

Proof of Claim. There are three cases for the nature of this revolt.

- (I) The revolt consists of all  $\alpha$ -type agents.

By Lemma 11, we know that, in either state, the probability that the expected number of  $\alpha$ -type agents differs from the expected number of  $\alpha$ -type agents by a multiplicative factor of  $\epsilon$  is  $\frac{1}{\exp(\Omega(\sqrt[3]{n}))}$ . Therefore, the probability that a revolt of size  $\mu^* - \epsilon$  is supported in both states is at least  $1 - \frac{1}{\exp(\Omega(\sqrt[3]{n}))}$ .

- (II) The revolt consists of all  $\alpha$ - and all  $\chi$ -type agents.

In this case, the presence of  $\chi$ -type agents slightly complicates the analysis. Actual  $\chi$ -type agents have slightly easier thresholds to satisfy ( $p$  and  $\mu$ ) than the  $\chi$ -type agents considered by the algorithms ( $p'$  and  $\mu'$ ). Our algorithms determined that, in expectation, agents with  $p'$  and  $\mu'$  thresholds would feel secure enough to revolt. Each actual  $\chi$ -type agent, then, must also feel secure enough to revolt, not just in expectation: By Lemma 11, the probability that the actual number each of  $\alpha$ -type and  $\chi$ -type agents differs from its respective expectation by a multiplicative factor of  $\frac{\epsilon}{6}$  is  $\frac{1}{\exp(\Omega(\sqrt[3]{n}))}$ . Combining these,

then,  $\chi$ -type agents believe with probability  $1 - \frac{1}{\exp(\Omega(\sqrt[3]{n}))} \geq p$  that at least  $\mu' - \frac{\epsilon}{3} = \mu$  agents feel secure enough to revolt, and are thus themselves secure enough to revolt. Applying Lemma 11 (from the perspective of Algorithm 2 this time, not the perspective of  $\chi$ -type agents) we again incur two error terms of  $\frac{\epsilon}{6}$  (one each for the  $\chi$ - and  $\alpha$ -type agents). Thus, we can conclude the probability that a revolt of size  $\mu^* - \epsilon$  is supported in both states is at least  $1 - \frac{1}{\exp(\Omega(\sqrt[3]{n}))}$ .

(III) The revolt consists of all  $\alpha$ -type agents and all candidate agents.

In the final case, we consider candidate agents instead of all agents of type  $\chi$ , and proceed exactly as we did in the second case, using Lemma 11 to conclude that the number of  $\alpha$ -type agents concentrate and Lemma 12 to conclude that the number of candidate agents concentrates. Once again, the result is that the probability that a revolt of size  $\mu^* - \epsilon$  is supported in both states is at least  $1 - \frac{1}{\exp(\Omega(\sqrt[3]{n}))}$ .  $\triangleleft$

$\triangleright$  **Claim.** Suppose that, in instance (1), the result is **A**. Then, the probability that a revolt of size  $\mu^* - \epsilon$  is supported in both states is at least  $\Pr[\text{State} = A] - \frac{1}{\exp(\Omega(\sqrt[3]{n}))}$ .

**Proof of Claim.** The proof of this claim is analogous to the previous proof, where events that are assigned probability at least  $1 - \frac{1}{\exp(\Omega(\sqrt[3]{n}))}$  are instead assigned probability at least  $\Pr[\text{State} = A] - \frac{1}{\exp(\Omega(\sqrt[3]{n}))}$ .  $\triangleleft$

The proof of the analogous claims for (2) are themselves analogous to the above cases for (1).

Finally, we note here that when  $\chi$ -type agents decide whether or not their thresholds are satisfied, they are not solely relying on their estimates of the fractions of different kinds of agents, as we describe above. They have additional knowledge, since they see the realized types of the agents in their context. However, for small graphs,  $\delta(n)$  can be chosen to account for this. As the graph grows, since each agent has at most  $O(\sqrt[3]{n})$  neighbors, the consequences of observing the types of a few adjacent agents is negligible after conditioning on the state. As a result, for large enough  $n$ , after the agent reasons about the state, the probabilistic effect of the agent viewing the types in her context is subsumed by the  $\epsilon$  error term. Furthermore, the appropriate choice of  $\delta(n)$  also accounts for the  $\frac{1}{\exp(\Omega(\sqrt[3]{n}))}$  terms present in each of the claims stated above, for each  $n$ .  $\blacktriangleleft$

**Proof of Theorem 9.** It follows from Lemma 10 that Algorithm 3 terminates in polynomial time with respect to the inputs  $G$  and  $P$ . We also note that Algorithm 3 terminates in polynomial time with respect to  $\frac{1}{\epsilon}$  and  $\frac{1}{\delta}$ .

Further, it follows from Lemma 13 that when Algorithm 3 outputs a case, that case is always true. It only remains to show that when exactly one case in the statement of PROMISE REVOLT is true, then Algorithm 3 necessarily outputs that case:

Here, the key is our use of three different potential revolt sizes ( $\mu^* - \epsilon$ ,  $\mu^*$ , and  $\mu^* + \epsilon$ ) and 3 different priors ( $P^-$ ,  $P$ , and  $P^+$ ) in defining the three cases of PROMISE REVOLT. By promising that exactly one of those three cases is true, we guarantee that the values of  $\mu$  and  $p$  are sufficiently far – distance at least  $\epsilon$  for  $\mu$  and distance at least  $\delta$  for  $p$  – from the crucial decision thresholds in Algorithm 1 (e.g. the value  $e_B(\chi \cup \alpha)$ , which is used to determine whether or not  $B$  is a candidate state)<sup>3</sup>. This ensures that both calls to Algorithm 1 will return the same values.

<sup>3</sup> Whether or not a decision threshold is in the set of *crucial* decision thresholds – the thresholds from which our promise guarantees  $p$  and  $\mu$  are sufficiently far – depends on  $\mu^*$ . In addition to  $e_B(\chi \cup \alpha)$ ,

We can illustrate this counterfactually:

Let  $\mu = e_B(\chi \cup \alpha) + \frac{\epsilon}{2}$  with  $e_B(\chi \cup \alpha) > \mu^* > e_B(C_C \cup \alpha)$  and let  $e_A(C_C \cup \alpha) > \mu + \frac{\epsilon}{2}$ .

Then,  $A$  and  $B$  would both be candidate states under the prior  $P^-$ . As a result, Algorithm 1 (run with prior  $P^-$ ) would return  $X_A = e_A(\chi \cup \alpha)$  and  $X_B = e_B(\chi \cup \alpha)$ . Note that  $X_A > X_B > \mu^*$ , so Algorithm 2 would return  $\Omega$  when run with inputs  $X_A$ ,  $X_B$ , and  $\mu^*$ . The same analysis holds for  $P^-$  with  $\mu$  and  $p$  incremented by  $\frac{\epsilon}{3}$  and  $\frac{\delta}{3}$ , respectively, so applying Lemma 13 implies that the case  $\Omega$  is true.

On the other hand, only  $A$  would be a candidate state under the prior  $P$ . Algorithm 1 run with the prior  $P$  would return  $X_A = e_A(C_C \cup \alpha)$  and  $X_B = e_B(C_C \cup \alpha)$ . Run with these inputs (and  $\mu^*$ ), Algorithm 2 would return  $\mathbf{A}$ . The same analysis holds for  $P$  with  $\mu$  and  $p$  incremented *and* decremented by  $\frac{\epsilon}{3}$  and  $\frac{\delta}{3}$ , which by Lemma 13 implies that the case  $\mathbf{A}$  is true. We can conclude that these values of  $\mu$  and  $\mu^*$  must be excluded by our promise for any input to PROMISE REVOLT for which our assumed constraints hold.

An argument similar to this counterfactual argument suffices to exclude any value of  $p$  or  $\mu$  that is insufficiently far from a crucial decision threshold in Algorithm 1 (along with associated constraints on  $\mu^*$ ) present in an instance of PROMISE REVOLT. ◀

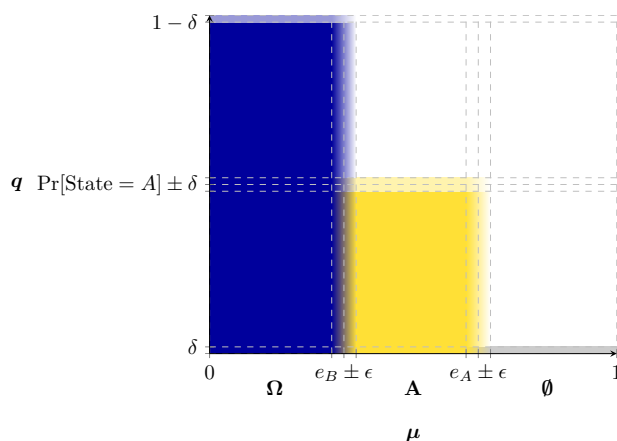
Now we can discuss the information that we gain from solving PROMISE REVOLT. In doing so, it is useful to refer to Figure 1, which contains the following illustration: Given  $\mu^*$ , we identify all values of  $q$  for which a revolt of size  $\mu^*$  is supported with probability at least  $q$  (dark blue and yellow regions), all values of  $q$  for which revolt of size  $\mu^*$  may be supported with probability  $q$  (light blue, yellow, and grey regions), and all values of  $q$  for which a revolt of size  $\mu^*$  is supported with probability strictly less than  $q$  (white regions). The blurry regions between distinct colors represent the inputs to PROMISE REVOLT for which two of the cases would overlap (and are therefore excluded from the set of inputs by the “promise”).

If we could perfectly solve REVOLT, then we would be able to perfectly define the boundaries – there would be no blurry regions between distinct colors, as in Figure 1. The shape of the resulting figure would characterize the equilibria of the network revolt game in the following sense: The blue column (corresponding to values of  $\mu^*$  for which only case  $\Omega$  of PROMISE REVOLT is true) would show the sizes of revolts that are supported in equilibrium with high probability regardless of the state, the next yellow and white column (corresponding to values of  $\mu^*$  for which only case  $\mathbf{A}$  is true) would show the sizes of revolts that are supported in equilibrium with high probability given that the state is  $A$ , and the last grey and white column (corresponding to values of  $\mu^*$  for which case  $\emptyset$  is true) would show the sizes of revolts that, with high probability, are not supported in equilibrium (or, equivalently, the sizes of revolts that are supported in equilibrium with low probability).

Although we cannot perfectly solve REVOLT, as shown in Figure 1, solving PROMISE REVOLT with Algorithm 3 allows us to approximate this shape. By choosing  $\epsilon$  to be very small, we can make the boundary cases only relevant for very small subsets of possible values of  $\mu^*$  and values of  $\mu$  in the prior  $P$ . And, as we have shown via the proof of Theorem 9, as  $n$  grows,  $\delta(n)$  quickly becomes very small. Consequently, the range of possibly forbidden values of  $p$  in the prior  $P$  also quickly becomes small and the probabilities for each class of distinct equilibria described above (when they exist), converge to 1,  $\Pr[\text{State} = A]$ , and 0, respectively.

---

it can also include  $e_A(C_C \cup \alpha)$  and the value of  $p$  below which  $e_A(C_C \cup \alpha) \geq \mu$  holds for candidate contexts and above which  $e_A(C_C \cup \alpha) < \mu$ .



■ **Figure 1** Illustrating what we learn from solving PROMISE REVOLT in an archetypal case. Here,  $e_B$  and  $e_A$  denote the expected size of the revolt supported in states  $B$  and  $A$ , respectively.

Lastly, we note that an interesting and surprising corollary to our analysis above is that Algorithms 1, 2, and 3 never use any information about the graph beyond the *degree sequence* of  $G$  – an anonymous list that records the degree of each vertex in the graph. That is, our algorithms require a list of the degrees of the agents, but do not require that any degree value be labelled with the identity of any agent, nor do they require any further information about the set of edges in the graph. Rather, the results of the algorithms are valid for any graph that is consistent with the provided degree sequence, because the concentration of the fraction of candidate agents and agents of each type supersedes the additional structure imposed by any concrete edge set consistent with the degree sequence.

We record this fact with the following proposition:

► **Proposition 14.** *To compute PROMISE REVOLT in polynomial time, we only require the degree sequence of the graph  $G$ ; the graph itself is not required.*

## 5 Broadening the Setting

The core intuition from our analysis above actually applies more generally than just to the setting of Section 4.1. In the full version of the paper, we explore settings where the value of interest is the size of the smallest supported revolt in each state (as opposed to the largest) and where there is an arbitrary number of states.

Here, we only discuss the most fundamental of the more general settings – extending our result to general graphs. In doing so, we focus on changes to Algorithm 1; the subsequent changes required to adapt Algorithms 2 and 3 are straightforward.

### 5.1 General Graphs

Suppose that we impose no restriction on the degree of the vertices in  $G$  in the statement of Theorem 9. It turns out that our results still hold. The existence of high-degree  $\chi$ -type vertices (i.e. those that would not exist when we assume an  $O(\sqrt[3]{n})$  bound on the maximum degree of any vertex), complicates the analysis, but only in the case where  $A$  is the only

candidate state. When  $A$  and  $B$  are both candidate states, we only need to calculate the expected number of all  $\chi$ -type agents, regardless of their degree. When there are no candidate states,  $\chi$ -type vertices are irrelevant.

When  $A$  is the only candidate state, however, there is a key difference: The presence of high-degree agent contexts in the set of candidate contexts would affect our earlier analysis using the concentration bound for the sum of random variables with limited dependence based on the fractional chromatic number of the constraint graph of the random variables (in the proof of Lemma 12), because the contexts of high-degree agents are correlated with many other agents' contexts.

Because of this, though, high-degree agents (with degree at least  $c \cdot \sqrt[3]{n}$  for sufficiently large  $c$ ) have a unique perspective on the graph; their contexts contain a significant amount of information that they can use in determining the state. More concretely, suppose that agent  $i$  is a high-degree agent and let  $N(i)$  be the set of neighbors of  $i$  in  $G$ .

Let  $X = \sum_{j \in N(i)} X_j$ , where  $X_j$  is an indicator random variable that is 1 when agent  $j$  is of type  $\alpha$  or type  $\chi$  and 0 otherwise.

Now, for any  $\epsilon_0 > 0$ , applying a standard Chernoff-Hoeffding bound, we have:

$$\Pr [|X - \mathbb{E}[X]| \geq \epsilon_0 (c \cdot \sqrt[3]{n})] \leq 2 \exp \left( \frac{-2(\epsilon_0 c)^2 n^{\frac{2}{3}}}{n^{\frac{1}{3}}} \right) \in \frac{1}{\exp(\Omega_{\epsilon, P}(\sqrt[3]{n}))}. \quad (1)$$

In particular, if we choose  $\epsilon_0$  such that  $|\mathbb{E}[X|\text{State} = A] - \mathbb{E}[X|\text{State} = B]| > 2\epsilon_0 (c \cdot \sqrt[3]{n})$ , then agent  $i$  (and by the same argument, any high-degree agent) can correctly determine the state with enough accuracy that their error can be absorbed into the  $\delta$  error term with an appropriate choice of  $\delta(n)$ . As a result, when the state is  $A$ , all high-degree  $\chi$ -type agents will behave like candidate agents (recall that  $A$  is a candidate state).

However, for computational purposes, we will not lump them in with the low-degree candidate agents, because Lemma 12 can only apply to low-degree candidates. Instead, in the first **if** statement after the condition that  $A$  is the only candidate state (the line in Algorithm 1 that reads “**if**  $e_A(C_C \cup \alpha) \geq \mu$  **then**”), we calculate  $e_A(C_C \cup \alpha \cup H_\chi)$  instead of just  $e_A(C_C \cup \alpha)$ , where  $H_\chi$  refers to the set of high-degree  $\chi$ -type vertices. If this is at least  $\mu$ , then when we calculate  $X_A$ , we include  $H_\chi$ . However, when we calculate  $X_B$ , we only include  $H_\chi$  if  $e_B(C_C \cup \alpha \cup H_\chi) \geq \mu$ , since those high-degree agents will not  $p$ -believe that the state is  $A$  regardless of the state the way that (low-degree) candidate agents will. If  $e_B(C_C \cup \alpha \cup H_\chi) < \mu$ , we calculate  $X_B = e_B(C_C \cup \alpha)$ , as described in Algorithm 1.

Next, we must show it is possible for the agents (and the algorithm) to accurately calculate the expected number of high-degree agents of type  $\chi$  and know that the actual number of such agents sufficiently concentrates around that expected number. Here, we can rely on the fact that the agents and the algorithm know the degree sequence of the graph. There is some subtlety involved: If the number of high-degree agents is small – less than  $\epsilon n$  – then we cannot provide a very useful concentration bound for the number of high-degree agents of type  $\chi$ . However, we can essentially ignore the high-degree agents in this case and absorb the error we incur by ignoring them into our choice of  $\delta(n)$ .

On the other hand, if there are at least  $\epsilon n$  high-degree agents, we can again use a standard Chernoff-Hoeffding bound to conclude that the actual number of high-degree  $\chi$ -type agents will, with high probability, be close to the expected number of such agents. The error here will be smaller than the error from Lemma 12 and so can be absorbed there.

Finally, there is one additional subtlety we must address for high-degree agents. While it is true that, as previously mentioned, their contexts contain a significant amount of information that they can use in determining the state, their contexts actually contain more than just

information about the state – they contain information about the actual realization of types for a significant number of agents. This point is the primary conceptual reason to make the distinction between high-degree and low-degree agents in the first place. Consequently, we need to show that high-degree agents tend to behave as if they only knew the state, when in fact it is possible that the number of agents of a certain type in their context differs greatly from its expectation and as a result they have additional information to use beyond the state of the world. For this, we again appeal to our previous argument that resulted in the bound expressed by the inequality (1), above.

That argument demonstrates that it is highly improbable for the information in a high-degree agent’s context to contradict what their belief would be solely given knowledge of the state. For example, if a high-degree agent uses her context to determine that the state is  $A$ , with high probability it will not also be the case that the context that she uses to make that determination has (far) fewer agents of any type than would be expected given that the state is  $A$ . Consequently, a high-degree agent will tend to act as if she is just calculating expectations and acting off of them (like a low-degree agent would), even though in actuality she has quite a bit of additional information.

## 6 Discussion

We proposed that the notion of common belief could be relaxed to a notion of factional belief in order to be more suited to social network settings and gave a natural definition of factional belief, drawing heavily from previous work on common belief. We then applied this definition theoretically and experimentally in a setting inspired by prior work about common knowledge and revolt games on networks to show how this definition moves beyond the limitations of previous work by being applicable in general graphs.

### 6.1 Open Questions and Future Work

The most clear direction for future work is to continue to apply factional belief in new settings and use it as a tool to understand strategic coordination and cooperation on networks. In particular, as mentioned in the introduction, the work of Stephen Morris provides many examples of applying common belief as a tool for gaining insight into a diverse range of settings. We believe that applying factional belief can yield similar results in social network settings, and that this paper represents an initial step in that process.

Additionally, though, there are some more subtle technical open questions regarding our definition of factional belief that are also worth exploring in future work:

1. Considering our definition of factional belief from the perspective of an infinite hierarchy of reasoning, akin to the initial definition for common knowledge that we provided (3), with regard to the  $\mu$  fraction of agents at each step in the hierarchy, it is not required by our definition that this be the same  $\mu$  fraction of agents at each step in the hierarchy. However, it is not clear whether or not this should always be the case. Is it possible to describe an event that is common  $(p, \mu)$ -belief, but for which the infinite hierarchy refers to a different  $\mu$  fraction of agents at some step? If so, what are the consequences of this for strategic coordination and cooperation?
2. It is not too difficult to modify the setting described in Section 4 to create a model where the underlying event that supports revolt does not necessarily neatly reduce to a single event that is common  $(p, \mu)$ -belief. For example, if there are multiple types of conditionally-revolting agents with different  $p$  and  $\mu$  thresholds, the event supporting revolt is more like a  $\mu$  fraction of agents believe with sufficient probability that their

thresholds are satisfied. This event encompasses common  $p$ -beliefs among certain agents of the same type regarding their  $\mu$  thresholds, but not precisely common  $(p, \mu)$ -beliefs. Is there a more general definition of factional belief that allows for the existence of different thresholds for different types of agents to still be encompassed in a single event that is a factional belief among all of the agents who feel satisfied enough to revolt?

We believe that answering these questions could yield further insight into factional belief that could inform its application in other settings.

---

## References

- 1 Robert J. Aumann. Agreeing to disagree. *The Annals of Statistics*, 4(6):1236–1239, 1976. URL: <http://www.jstor.org/stable/2958591>.
- 2 Cristina Bicchieri. *The Grammar of Society: The Nature and Dynamics of Social Norms*. Cambridge University Press, 2005. doi:10.1017/CB09780511616037.
- 3 Michael Suk-Young Chwe. Structure and strategy in collective action. *American Journal of Sociology*, 105(1):128–156, 1999. doi:10.1086/210269.
- 4 Michael Suk-Young Chwe. Communication and Coordination in Social Networks. *The Review of Economic Studies*, 67(1):1–16, January 2000. doi:10.1111/1467-937X.00118.
- 5 Michael Suk-Young Chwe. *Rational Ritual: Culture, Coordination, and Common Knowledge*. Princeton University Press, 41 William Street, Princeton, New Jersey 08540, 2013.
- 6 Benjamin Golub and Stephen Morris. Expectations, networks, and conventions, 2017. doi:10.2139/ssrn.2979086.
- 7 Benjamin Golub and Stephen Morris. Higher-order expectations, 2017. doi:10.2139/ssrn.2979089.
- 8 Dov Monderer and Dov Samet. Approximating common knowledge with common beliefs. *Games and Economic Behavior*, 1(2):170–190, 1989. doi:10.1016/0899-8256(89)90017-1.
- 9 Stephen Morris. Approximate common knowledge revisited. *International Journal of Game Theory*, 28(3):385–408, August 1999. doi:10.1007/s001820050116.
- 10 Stephen Morris. Contagion. *The Review of Economic Studies*, 67(1):57–78, January 2000. doi:10.1111/1467-937X.00121.
- 11 Stephen Morris. Coordination, Communication, and Common Knowledge: A Retrospective on the Electronic-mail Game. *Oxford Review of Economic Policy*, 18(4):433–445, December 2002. doi:10.1093/oxrep/18.4.433.
- 12 Stephen Morris. Coordination, timing and common knowledge. *Research in Economics*, 68(4):306–314, 2014. doi:10.1016/j.rie.2014.04.004.
- 13 Stephen Morris and Hyun Song Shin. Approximate common knowledge and co-ordination: Recent lessons from game theory. *Journal of Logic, Language and Information*, 6(2):171–190, April 1997. doi:10.1023/A:1008270519000.
- 14 Stephen Morris, Hyun Song Shin, and Muhamet Yildiz. Common belief foundations of global games. *Journal of Economic Theory*, 163:826–848, 2016. doi:10.1016/j.jet.2016.03.007.
- 15 Ariel Rubinstein. The electronic mail game: Strategic behavior under “almost common knowledge”. *The American Economic Review*, 79(3):385–391, 1989. URL: <http://www.jstor.org/stable/1806851>.



# Tiered Random Matching Markets: Rank Is Proportional to Popularity

**Itai Ashlagi**

Department of Management Science and Engineering, Stanford University, CA, USA  
iashlagi@stanford.edu

**Mark Braverman**

Department of Computer Science, Princeton University, NJ, USA  
mbraverm@cs.princeton.edu

**Amin Saberi**

Department of Management Science and Engineering, Stanford University, CA, USA  
saberi@stanford.edu

**Clayton Thomas**

Department of Computer Science, Princeton University, NJ, USA  
claytont@cs.princeton.edu

**Geng Zhao**

Department of Computer Science, Stanford University, CA, USA  
gengz@stanford.edu

---

## Abstract

We study the stable marriage problem in two-sided markets with randomly generated preferences. Agents on each side of the market are divided into a constant number of “soft” tiers, which capture agents’ qualities. Specifically, every agent within a tier has the same public score, and agents on each side have preferences independently generated proportionally to the public scores of the other side.

We compute the expected average rank which agents in each tier have for their partners in the man-optimal stable matching, and prove concentration results for the average rank in asymptotically large markets. Furthermore, despite having a significant effect on ranks, public scores do not strongly influence the probability of an agent matching to a given tier of the other side. This generalizes the results by Pittel [20], which analyzed markets with uniform preferences. The results quantitatively demonstrate the effect of competition due to the heterogeneous attractiveness of agents in the market.

**2012 ACM Subject Classification** Theory of computation → Algorithmic game theory and mechanism design

**Keywords and phrases** Stable matching, stable marriage problem, tiered random markets, deferred acceptance

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.46

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/2009.05124>.

**Funding** *Mark Braverman*: Research supported in part by the NSF Alan T. Waterman Award, Grant No. 1933331, a Packard Fellowship in Science and Engineering, and the Simons Collaboration on Algorithms and Geometry. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the author and do not necessarily reflect the views of the National Science Foundation.

## 1 Introduction

The theory of stable matching, initiated by Gale and Shapley [9], has led to a deep understanding of two-sided matching markets and inspired successful real-world market designs. Examples of such markets include marriage markets, online dating, assigning students to



© Itai Ashlagi, Mark Braverman, Amin Saberi, Clayton Thomas, and Geng Zhao;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 46; pp. 46:1–46:16



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

schools, labor markets, and college admissions. In a market matching “men” to “women” (a commonly used analogy), a matching is stable if no man-woman pair prefer each other over their assigned partners.

A fundamental issue is characterizing stable outcomes of matching markets, i.e. the outcome agents should *expect* based on market characteristics. Such characterizations are not only useful for describing outcomes but also likely to be fruitful in market designs. Numerous papers so far have studied stable matchings in random markets, in which agents’ preferences are generated uniformly at random [20, 16, 3, 22]. This paper contributes to the literature by expanding these results to a situation where preferences are drawn according to different tiers of “public scores”, generalizing the uniform case. We ask how public scores, which correspond to the attractiveness of agents, impact the outcome in the market.

Formally, we study the following class of *tiered random markets*. There are  $n$  men and  $n$  women. Each side of the market is divided into a constant number of “soft tiers”. There is a fraction of  $\epsilon_i$  women in tier  $i$ , each of which has a public score  $\alpha_i$ . And there is a fraction of  $\delta_j$  men in tier  $j$ , each of which has a public score  $\beta_j$ . For each agent we draw a complete preference list by sampling without replacement proportionally to the public scores of agents on the other side of the market.<sup>1</sup> So a man’s preference list is generated by sampling women one at a time without replacement according to a distribution that is proportional to their public scores. Using  $\alpha, \epsilon$  to denote the vector of scores and proportions of tiers on the women’s side, we see that the marginal probability of drawing a woman in tier  $i$  is  $\alpha_i / (n\epsilon \cdot \alpha)$ . An analogous statement holds for the tier configuration  $\beta, \delta$  of the men. These preferences are a natural next-step beyond the uniform distribution over preference lists, and provide a priori heterogeneous quality of agents while still being tractable to theoretical analysis.

Our primary goal is to study the *average rank* of agents in each tier under the man-optimal stable matching, with a focus on the asymptotic behavior in large markets. The rank of an agent is defined to be the index of their partner on their full preference list, where lower is better. Additionally, we prove results on the *match type distribution*, i.e. the fraction of tier  $i$  women matched to tier  $j$  men (for each  $i, j$ ).

We show that, for large enough markets, the following hold to within an arbitrarily small approximation factor:

- (i) (Theorem 4.8.) With high probability, the average rank of men in tier  $j$  is

$$\frac{\epsilon \cdot \alpha}{\alpha_{\min}} \cdot \frac{1}{\delta \cdot \beta^{-1}} \cdot \frac{\ln n}{\beta_j}.$$

- (ii) (Theorem 5.1.) With high probability, the average rank of women in tier  $i$  is

$$(\delta \cdot \beta)(\delta \cdot \beta^{-1}) \frac{\alpha_{\min}}{\alpha_i} \frac{n}{\ln n}.$$

- (iii) (Theorem 5.2.) The probability that a woman in tier  $i$  matches to a man in tier  $j$  is  $\delta_j$ .

In the above,  $\beta^{-1} = \{1/\beta_j\}$  denotes the vector of the reciprocals of men’s public scores,  $\alpha_{\min}$  denotes the smallest public score on the women’s side, and  $\mathbf{x} \cdot \mathbf{y}$  denotes the dot product of the vectors  $\mathbf{x}$  and  $\mathbf{y}$ .

<sup>1</sup> These are also termed popularity-based preferences [10, 13] and also equivalent to generating preferences according to a Multinomial-Logit (MNL) induced by the public scores.

**Intuition and Observations.** As in the case of uniform preferences [20], in the man-optimal stable outcome, men get a much lower rank than women. Indeed, both men and women get the same order of rank as in the uniform case ( $\ln n$  and  $n/\ln n$ , respectively). This in itself is an interesting consequence of this work – a constant tier structure affects the market only up to constants. This fact also highlights that determining these constants is an interesting area for investigation, as the constants capture how the outcome of the market changes with respect to the public scores. The first observation we make is that agents on each side get a rank inversely proportional to their public score.

Perhaps more interesting is the following observation: The rank of both sides depends on the tier structure of the other side, but *each tier is affected the same amount* by the tier parameters of the other side. This is closely related to the fact that the probability of a woman matching to a man in tier  $j$  is proportional to only the number of men in tier  $j$  (regardless of the tier the woman lies in). Moreover, both  $\epsilon \cdot \alpha / \alpha_{\min}$  and  $(\delta \cdot \beta)(\delta \cdot \beta^{-1})$  are always greater than or equal to one<sup>2</sup>. Thus, in these markets, any heterogeneity in the public scores of one side harms the average ranks of the other side (but does not significantly affect the likelihood that an agent matches to a certain tier on the other side).

Another interesting feature is the following: While the average ranks for men's tiers depend on public score distributions on both sides of the market, the average rank of women in tier  $i$  depends only on the ratio between  $\alpha_i$  and the public score  $\alpha_{\min}$  of the bottom tier of women (and the distribution of public scores on the men's side). Intuitively, the rank of the men depends on the distribution of scores of the women because *men are competing to avoid being matched to the lowest tier of women*.

To elaborate on that last point, let us first consider the total number of proposals made during the man-proposing deferred acceptance process (DA). The algorithm will terminate when the last woman receives a proposal. Naturally one would expect that this woman will belong to the bottom tier. Therefore, using standard coupon collector arguments, the total number of proposals made to women *in the bottom tier* until they all receive a proposal is expected to be  $(\epsilon_{\min} n) \ln(\epsilon_{\min} n)$ , where  $\epsilon_{\min}$  is the fraction of women in the bottom tier. These proposals are a  $\epsilon_{\min} \alpha_{\min} / \epsilon \cdot \alpha$  fraction of the total proposals, so one expects the number of total proposals to be

$$\frac{(\epsilon_{\min} n) \ln(\epsilon_{\min} n)}{\epsilon_{\min} \alpha_{\min} / \epsilon \cdot \alpha} = \frac{\epsilon \cdot \alpha}{\alpha_{\min}} \cdot n \ln n - O(n).$$

This introduces the factor of  $\epsilon \cdot \alpha / \alpha_{\min}$  in result (i) on the men's ranks (i.e. the number of proposals per man).

On the other hand, the probability that one of these proposals goes to a woman in tier  $i$  is  $\alpha_i / (n \epsilon \cdot \alpha)$ , implying that such a woman should receive roughly  $(\alpha_i / \alpha_{\min}) \ln n$  proposals. Thus, for a given woman, the increase in the total number of proposals caused by the tier proportions  $\epsilon$  is exactly canceled out by the likelihood that a proposal goes to that woman, and the only thing that matters is the woman's score (relative to the bottom tier). If men are uniform, women should then expect rank roughly  $(\alpha_{\min} / \alpha_i)(n / \ln n)$ , which helps explain the corresponding factors in result (ii).

Consider now the public scores of the men, and for simplicity assume that the bottom tier of men has score 1. Suppose for the sake of demonstration that every time a man with public score  $\beta_j$  proposes to a woman who is already matched, this man is  $\beta_j$  times more likely to

<sup>2</sup> To prove  $(\delta \cdot \beta)(\delta \cdot \beta^{-1}) \geq 1$ , use Jensen's inequality to conclude that  $\sum_j \delta_j \beta_j \geq (\sum_j \delta_j \beta_j^{-1})^{-1}$ .

be accepted than a man with public score 1.<sup>3</sup> We would expect that such a man makes a  $1/\beta_j$  fraction fewer proposals before his next acceptance, and indeed  $1/\beta_j$  fewer proposals overall. Let  $S$  be the total number of proposals, let  $r_j$  denote the rank of a man in tier  $j$ , and  $r_{\min}$  the rank of the bottom tier of men. If every tier of size  $\delta_j n$  each accounts for a share of proposals proportional to  $1/\beta_j$ , then we should have

$$S = \sum_j (n\delta_j)\beta_j^{-1}r_{\min} \implies r_{\min} = \frac{S}{n\delta \cdot \beta^{-1}}, \quad r_j = \frac{S}{(n\delta \cdot \beta^{-1})\beta_j},$$

which introduces the factor of  $1/((\delta \cdot \beta^{-1})\beta_j)$  in result (i) on the men's rank.

The final remaining factor in our results is  $(\delta \cdot \beta)(\delta \cdot \beta^{-1})$  in result (ii). Deriving this term requires reasoning about the number of proposals from each tier of men received by a fixed woman  $w$ . Building from the previous paragraph, we reason that each of the  $\delta_j n$  men in tier  $j$  makes a number of proposals proportional to  $1/\beta_j$ . Each such proposal has the same probability of going to  $w$ , regardless of the tier  $j$ . So the number of proposals  $w$  receives from tier  $j$  men is proportional to  $\delta_j/\beta_j$ . The factor  $(\delta \cdot \beta)(\delta \cdot \beta^{-1})$  then arises for somewhat technical reasons (described in Section 5) which have to do with the way women generate their preference lists.

We now describe how result (iii), which may seem somewhat more mysterious than the other results, emerges as a corollary of computing the ranks women receive. We argued above that a woman  $w$  in tier  $i$  receives approximately  $(\delta_j/\beta_j)U_i$  proposals from men in tier  $j$ , for some value of  $U_i$  independent of  $j$ . Recall that  $w$  applies weight  $\beta_j$  to each proposal she sees from a man in tier  $j$ . Moreover, the identity of  $w$ 's favorite proposal is independent of the order in which  $w$  saw proposals. Thus, the probability that  $w$ 's favorite proposal (i.e. the proposal of the man she matches to) came from tier  $j$  is approximately  $(\delta_j U_i)/U_i = \delta_j$ , which is *independent of*  $\beta_j$ , as well as independent of the tier  $w$  is in. Thus, up to lower order terms, the distribution of match types is the same as it would be in a uniformly random matching market, and the match is not assortative.

Intuitively, result (iii) arises when men make enough proposals to offset any disadvantage (in the type of their match) they have due to public score. Due to the highly connected and relatively competitive nature of our markets, men in the lowest tier make more proposals, but they are not more likely to end up matched with lower tier agents. Put another way, men in lower tiers are less likely to attain matches they idiosyncratically like, but often settle for a high-quality agent which is low on their personal preference list. This indicates that public scores that differ by constant weight do not provide any significant a priori predictive power over the matches agents receive. In particular, agents with lower public scores can still hope to achieve high-tier matches if they consider enough options.

**Techniques.** Our proofs require developing some technical tools that may be of independent interest, especially when we reason about the ranks achieved by the men. We build on the analysis of DA from [23, 20, 13, 3] to handle public scores rather than just uniform random preferences. As in these previous works, a key step in our proof is letting all men but one (call him  $m$ ) first propose and match through DA, and then tracking the proposals of  $m$  (this works because DA is independent of the order of proposals). For demonstration purposes,

<sup>3</sup> As we discuss below, this approximation is only valid if the woman is already matched with a man she ranks highly. A major technical step in our proof is showing that, in certain situations, "enough" women are "matched well enough" for this approximation to be used.

let's call the proposals before man  $m$  the “setup”. A key fact in previous works is that the distribution of proposals made by  $m$  is identical for every man, and moreover that the distribution of setups is identical as well. This fails to hold in tiered random markets, and thus we must develop new techniques.

We prove that, for “most” setups, the rank a man can achieve is approximately given by a certain geometric distribution, whose parameter  $p$  is essentially the probability that a proposal by that man will be accepted. We then prove that, up to lower order terms, this success parameter scales up with the public score of the men. This gives the fact that the rank of men is inversely proportional to public score.

Characterizing the setups where our proof goes through requires a technical analysis, and we term the setups which work “smooth matching states”. The most crucial thing we need for these setups is that *many women are matched to partners they rank highly*, which helps us prove that 1) men are likely to remain matched to their first acceptance (so our approximation with a geometric distribution is valid), and 2) a man with fitness  $\beta$  is approximately  $\beta$  times more likely to be accepted every time. For details, see Section 4.

Finally, to prove that the average rank of men within a tier *concentrates*, we need to show the correlation between the ranks of different men is not too large. Thus, we track the proposals of the last *two* men to propose, and find that the joint distribution of the ranks of these men can be approximated by a pair of independent geometric distributions. Intuitively, this is because men do not propose to very many women overall, and thus the last two men are unlikely to interfere with each other as they make proposals.

The crucial aspects of our model are that preferences of each agent are independent and identically distributed, that preference weights are constant, and that the market is roughly in balance. While our techniques are useful to reason about markets which do not have these properties, the results are not nearly as clean; indeed the tier structure simplifies our analysis, but most of it goes through if each agent has an individual, constant, bounded public score.

## 1.1 Related literature

Several papers have studied matching markets with complete preference lists that are generated uniformly at random. Coupon collector techniques are used in [23] to upper bound the men's average rank by  $\ln n$ . The papers [20, 16, 21] analyze further balanced markets with  $n$  men and  $n$  women. They find that in the man-optimal stable matching in balanced markets, men and women match on average to their  $\ln n$  and  $\frac{n}{\ln n}$  ranks, respectively. Our results generalize these findings to markets with preferences induced by public scores, thus incorporating much more heterogeneity in the market.

Several papers study markets with uniformly drawn preferences and an imbalance between men and women ([3, 22, 6]). These papers find that in any stable matching the average ranks of men and women are similar to the average ranks under the short-side-proposing DA. Additionally, [14] investigates the relation between the imbalance and the length of preference lists (though the model is still uniform for each agent). This paper does not consider imbalanced markets but we believe that similar techniques to those we develop will be useful to reason about unbalanced tiered random markets.

Several papers look at random matching markets in which preferences are generated based on public scores [13, 17, 1]. These papers restrict attention to the size of the core (a measure of the difference between the man-optimal and woman-optimal outcome) and strategic manipulation of agents under a stable matching mechanism. Key assumptions in

these papers generate outcomes which leave many agents unmatched. In particular, their models either assume that preference lists of men are of constant length, or, alternatively, one side has many more agents than the other.<sup>4</sup>

Closely related to this paper is [10], which primarily studies a special case of highly correlated popularity preferences which is termed “geometric preferences”. While our work focuses on the rank agents achieve in the man-optimal outcome (a canonical stable matching), [10] focuses on the size of the core (more specifically, they study the number of stable partners that agents have in typical stable matchings) using techniques specialized to geometric preferences.

Other papers have addressed tiered matching markets, especially in market design settings. However, these papers mostly study “hard tiers”, i.e. such that agents in higher tiers are deterministically ranked above lower tiers by every agent on the other side. Examples include [4, 2]. [18] also considers a certain restricted tiered model of cardinal utilities (which is incomparable with our model), focusing on which tier of agents match to which tier.

Our contribution to the literature is a detailed study of “soft tiers”, a natural special case of the popularity preferences of [13, 17, 10]. In cases where each agent’s utility for each match on the other side is independent and identically distributed, popularity preferences are the natural next step beyond uniform markets, as they model situations where agents on each side have significant but non-definitive variation in a priori quality. Our techniques build on the large body of work analyzing the “proposal dynamics” of deferred acceptance for random preferences, such as [23, 13, 3, 10]. Our results give insight into how constant-factor preference biases affect stable matching markets, including the first explicit calculations of expected rank beyond uniform markets.

The rest of the paper is organized as follows: Section 2 offers basic definitions and preliminaries for our discussion. Section 3 studies the tiered coupon collector process, which serves as an important coupling process for the deferred acceptance algorithm. Section 4 and 5 present the core results of this paper, namely the average rank among tiers of men and women. For missing proofs, see the full version of this paper.

## 2 Definitions and Preliminaries

A matching market consists of a finite set of men  $M$  and a finite set of women  $W$ . Each man (woman) has a complete and strict preference list over women (men). A matching is a mapping  $\mu : M \cup W \rightarrow M \cup W$  such that: for every  $m \in M$ ,  $\mu(m) \in W$  (or  $\mu(m)$  is undefined), for every woman  $w \in W$ ,  $\mu(w) \in M$  (or  $\mu(w)$  is undefined), and for every  $m \in M$  and  $w \in W$ ,  $\mu(m) = w$  if and only if  $\mu(w) = m$ . A matching  $\mu$  is *stable* if no man-woman pair who are not matched in  $\mu$  prefer each other to their matched partners.

It is well-known that there is a unique man-optimal stable matching, which can be found using the man-proposing deferred acceptance algorithm (DA). While this algorithm does not fully specify an execution order, it is a classically known result that the order does not affect the final outcome.

► **Lemma 2.1** ([9, 19]). *The same proposals are made in every run of DA, regardless of which man is chosen to propose at each step.*

---

<sup>4</sup> Some papers additionally consider manipulations in more restricted randomized settings [7] or in deterministic (worst case) settings [11].

---

**Algorithm 1** (Man-Proposing) Deferred Acceptance Algorithm (DA).

---

```

1 Initialize matching  $\mu$  to be empty (i.e. every agent's partner is undefined);
2 Initialize  $\mathcal{U} = M$  to be the set of all unmatched men;
3 while  $|\mathcal{U}| > 0$  do
4   Choose any  $m \in \mathcal{U}$ ;
5   Let  $m$  propose to his most preferred woman  $w$  to whom he has not made a
   proposal yet;
6   if  $w$  prefers  $m$  to  $\mu(w)$  (or if  $\mu(w)$  is undefined) then
7     if  $\mu(w)$  is defined then Add  $\mu(w)$  to  $\mathcal{U}$ ;
8     Remove  $m$  from  $\mathcal{U}$ ;
9     Assign  $\mu(w) = m$ ;
10  end
11 end

```

---

We study the man-optimal stable matching in a class of tiered random markets, which will be defined below. We will assume that  $|M| = |W|$  and that no agent finds any other agent on the other side unacceptable. We will also assume that each side draws their preferences from an identical and independent underlying distribution, and moreover these preferences are generated by repeatedly sampling without replacement from a fixed distribution on the *agents* of each side. In [13, 10], this assumption is termed “popularity-based preferences”, with the weight of an agent in the distribution intuitively indicating their popularity for agents on the other side.

Our main goal is to study randomized matching markets with a *constant number of constant weight tiers* of agents on each side. For this entire paper, we consider the tier structure to be defined by fixed proportions  $\epsilon, \delta$  of agents in each tier and constant weights  $\alpha, \beta$  for each tier, and we investigate the outcome of the man-proposing DA as  $n \rightarrow \infty$ .

► **Definition 2.2.** Consider constant vectors  $\alpha, \epsilon \in \mathbb{R}_{>0}^{k_1}$  and  $\beta, \delta \in \mathbb{R}_{>0}^{k_2}$ , where  $\|\epsilon\|_1, \|\delta\|_1 = 1$ . A tiered matching market of size  $n$  with respect to  $\alpha, \epsilon, \beta, \delta$  is defined by generating agents’ preference lists as follows:

- The set of  $n$  women  $W$  is divided into tiers  $T_1, \dots, T_{k_1}$ , of size  $|T_i| = \epsilon_i n$  each<sup>5</sup>. Define a distribution  $\mathcal{W}$  on women such that a woman in tier  $i$  is selected with probability proportional to  $\alpha_i$ . That is, the weight of  $w \in T_i$  in  $W$  is  $\alpha_i / (n\epsilon \cdot \alpha)$  (which we often denote by  $\pi_i$ ).
- The set of  $n$  men  $M$  is divided into tiers  $T_1, \dots, T_{k_2}$ , of size  $|T_j| = \delta_j n$  each. Define a distribution  $\mathcal{M}$  on men such that a man in tier  $j$  is selected with probability proportional to  $\beta_j$ . That is, the weight of  $m \in T_j$  in  $M$  is  $\beta_j / (n\delta \cdot \beta)$ .

For each man  $m$  independently, women are repeatedly sampled from  $\mathcal{W}$  without replacement, and the order in which women are selected is  $m$ ’s preference list. Preferences for the women are analogously drawn over the distribution  $\mathcal{M}$ . The rank that a man has for a woman  $w$  is the index of  $w$  on his preference list (where lower is better).

---

<sup>5</sup> Note that, for most vectors  $\epsilon, \delta$ , many values of  $n$  will produce tier sizes which are not integers. However, as all our results are *continuous* in  $\epsilon, \delta$  this is not a problem – for any particular fixed  $n$ , each tier size can be rounded in a way that effectively just changes  $\epsilon, \delta$  by a tiny amount, and our results will still hold as written as  $n \rightarrow \infty$ .



We refer to each  $\alpha_i$  as the *weight* or *public score* of the women in tier  $i$ , and similarly for the men. For simplicity of certain arguments, we assume that each  $\alpha_i \geq 1$  and each  $\beta_j \geq 1$  (although for clarity of our results, we do not assume that the smallest weight is exactly 1). We write  $\alpha_{\min}$  for the weight of the bottom tier of women, and  $\epsilon_{\min}$  for the corresponding tier proportion.

Using a simple generalization of the “principle of deferred decisions” used in [15], we can arrive at a characterization of the random process of running DA with a tiered matching market.

► **Lemma 2.3.** *The distribution of runs of DA for a tiered matching market can be generated as follows: For the men, every time a man is chosen to propose, he samples a woman at random from  $\mathcal{W}$ , and repeats this until he samples a woman who he has not yet proposed to.*

*For the women, suppose  $w$  has seen proposals from a set of men  $p(w)$ , and let  $\Gamma_w = \sum_{m \in p(w)} \beta(m)$ , where  $\beta(m)$  denotes the public score of a man  $m \in p(w)$ . Then if a proposal from a man  $m_*$  with public score  $\beta_*$  arrives,  $w$  accepts the proposal from  $m_*$  with probability*

$$\frac{\beta_*}{\beta_* + \Gamma_w}.$$

**Proof.** The above formula gives the probability that  $m_*$  is chosen as  $w$ ’s favorite out of the set of men  $p(w) \cup \{m_*\}$ . The only additional observation we need to make is that the probability that  $m_*$  is the new favorite is independent of the identity of the old favorite. ◀

We often call  $\Gamma_w$  the total “weight of proposals” woman  $w$  has seen at some point during DA.

## 2.1 Deferred acceptance with re-proposals

With respect to any popularity-based model of preferences, we can define a procedure analogous to DA. In our case, we will show that the difference between DA and this procedure is indeed small.

► **Definition 2.4.** *Consider any random matching market with men’s preferences determined by sampling from a distribution  $\mathcal{W}$  over women. The deferred acceptance with re-proposals algorithm is defined as being identical to Algorithm 1, except*

- *Every time a man is chosen to propose to a woman, he draws a woman from  $\mathcal{W}$  with replacement, and may propose more than once to a single woman.*
- *Women’s preferences are consistent throughout proposals from the same man (so if a woman rejected a man before, she will reject him again).*

Since re-proposals are ignored, this process will always yield the same outcome as algorithm 1.

**Notation.** We write  $x = (1 \pm \epsilon)y$  to mean  $(1 - \epsilon)y \leq x \leq (1 + \epsilon)y$ . We let  $\epsilon$  denote an arbitrarily small constant greater than 0, while  $\epsilon$  and  $\epsilon_i$  denote the tier parameters of the women. We let  $\alpha_{\min}$  denote the smallest public score for the women’s side, and  $\epsilon_{\min}$  denotes the corresponding tier proportion. We let  $\mathbf{v} \cdot \mathbf{w}$  denote the inner product of vectors  $\mathbf{v}, \mathbf{w}$ . We denote the exponential and geometric distributions by  $\text{Exp}(\lambda)$  and  $\text{Geo}(p)$ , respectively. We denote the fact that a random variable  $X$  is a draw from a distribution  $D$  by  $X \sim D$ . We use  $X \preceq Y$  to denote the fact that  $X$  is statistically dominated by  $Y$  (i.e. for all  $t \in \mathbb{R}$ , we have  $\mathbb{P}[X \geq t] \leq \mathbb{P}[Y \geq t]$ ). We let  $\text{Cov}(X, Y)$  denote the covariance of  $X$  and  $Y$ . We write  $f(n) = \tilde{O}(g(n))$  if there exists a constant  $k$  such that  $f(n) = O(g(n) \log^k(g(n)))$ .

### 3 The Coupon Collector and the Total Number of Proposals

Fix a tier structure  $\alpha, \epsilon$  corresponding to men's preferences over the women. Consider running deferred acceptance with re-proposals. Recall that each man samples a woman in tier  $i$  with probability  $\pi_i = \alpha_i / (n\epsilon \cdot \alpha)$  each draw. Define  $\pi_{\min} = \alpha_{\min} / (n\epsilon \cdot \alpha)$  as the probability of drawing a woman in the lowest tier (and keep in mind that  $\pi_{\min}$  scales like  $O(1/n)$ ).

The core tool we use to reason about the total number of proposals in DA is the classically studied coupon collector process. In particular, we study this process when coupons from different tiers are drawn with a constant-factor difference in probability.

► **Definition 3.1.** *Given a probability distribution  $(p_i)_{i \in [n]}$ , we define the coupon collector with unequal probabilities as follows: once every time step, an integer from  $[n]$  is drawn independently and with replacement according to distribution  $(p_i)_{i \in [n]}$ . The coupon collector random variable with respect to  $(p_i)_{i \in [n]}$  is defined as the number of total draws required before every integer in  $[n]$  has appeared at least once.*

*The coupon collector  $T$  which we are interested in is defined by taking the distribution  $\mathcal{W}$  of men's preferences.*

We will show in Section 3.1 that, in our case, this random process is also very close to that of DA (without re-proposals). For now, we simply bound the expectation of the coupon collector (with the proof deferred to the full version). Note that similar probabilistic problems have been considered before (see e.g. [5, 8]) but we include our own full proofs in the full version for completeness.

► **Theorem 3.2.** *Let  $T$  denote the number of draws in a coupon collector process with weights proportional to  $\mathcal{W}$ . We have*

$$\mathbb{E}[T] = (1 \pm O(1/\ln n)) \frac{\epsilon \cdot \alpha}{\alpha_{\min}} n \ln n.$$

► **Remark 3.3.** While we are mostly interested in the asymptotic performance of these matching markets, we make one comment here that the above big- $O$  notation hides a constant factor of order  $\ln(1/\epsilon_{\min})$ . For small values of  $\epsilon_{\min}$ , this can be much larger than  $\ln n$  for most realistic market sizes. Note that this error term already showed up in the intuition given in Section 1, where our estimate for the total number of proposals had an additive term of  $O(\ln(\epsilon_{\min})n)$ . For more information, see the discussion of coupon collector lower bound in the full version of this paper.

#### 3.1 The Total Number of Proposals in Deferred Acceptance

Let  $S = S_n$  denote the total number of proposals made a run of DA with random preferences given by our tiered market. As before, let  $T = T_n$  denote the distribution of a coupon collector with distribution  $\mathcal{W}$ . As in many prior studies of randomized deferred acceptance, our starting point is the fact that  $S$  is statistically dominated by  $T$ :

The connection to stable matchings is the following very simple observation, which has been used in many previous works [16, 20]:

► **Proposition 3.4.** *The coupon collector random variable  $T$  is distributed identically to the total number of proposals made in deferred acceptance with re-proposals (regardless of the preferences that women have for men).*

*Moreover, if  $S$  is the number of proposals in DA, then  $S \preceq T$  (i.e.  $S$  is statistically dominated by  $T$ ).*

**Proof.** First, recall that DA terminates as soon as every man is matched. Observe that women never return to being unmatched once they receive a single proposal. Because the market is balanced (i.e.  $|W| = |M|$ ), this means DA will terminate as soon as every woman has been proposed to. Moreover, because re-proposals are allowed, every proposal is distributed exactly according to  $\mathcal{W}$ . Thus, ignoring the identity of the man doing the proposing,  $T$  is distributed exactly according to the coupon collector random process.

Furthermore, we can recover the exact distribution  $S$  of proposal in DA simply by ignoring each repeated proposal in  $T$ . Thus,  $S \leq T$  for each run of deferred acceptance with re-proposals, so  $S \preceq T$ .  $\blacktriangleleft$

We proceed to show that the upper bound provided by  $T$  is essentially tight, i.e. there is not a big difference between  $T$  and  $S$ . The key step will be to upper bound maximum number of distinct women any man proposes to in  $S$ , and thus upper bound the probability that any proposal in  $T$  is a repeat for the man making the proposal. Crucially, this lemma will have to account for the preferences of the women (which up until this point have been ignored, but which play a significant role in the distribution of proposals in DA). Recall that we denote the sizes of the tiers of the men by the vector  $\delta$ , and the public scores of the men in each tier by  $\beta$ .

► **Lemma 3.5.** *Consider running DA with all men except  $m_*$ , and suppose that at most  $O(n \ln n)$  proposals are made during this process. Afterwards, consider  $m_*$  joining and run DA until the end. Then for any  $C \geq 0$ , with probability  $1 - 1/n^C$ , the number of proposals made by  $m_*$  is at most  $O(C \ln^2 n)$ .*

**Proof.** This proof follows a similar logic as the proof of Lemma B.4 (ii) in [3]. Suppose  $m_*$  has public score  $\beta_*$ , and that he proposes at the end (and  $O(n \ln n)$  prior proposals have been made). We proceed as follows:

1. When  $m_*$  makes a proposal, he will choose a woman who he has not yet proposed to. For some fixed proposal index  $i$  of  $m_*$ , let's denote the set of all women  $m_*$  has not proposed to by  $W_*$ , and denote by  $\mathcal{W}_*$  the distribution of  $m_*$ 's next proposal, i.e. a sample over  $W_*$  weighted by the public scores  $\alpha_i$ . For a women  $w$  denote her sample weight by  $\alpha(w)$  and the set of proposals she has received by  $p(w)$ . Further denote by  $\Gamma_w = \sum_{m \in p(w)} \beta(m)$  the sum of the public scores of men who have proposed to  $w$ . Suppose that  $|W_*| \geq n/2$ , i.e. that  $m_*$  has not yet proposed to over half the women. Using the assumption that the total number of proposals made is at most  $O(n \ln n)$ , we can bound the expected total weight of proposals women have seen by

$$\mathbb{E}_{w \sim \mathcal{W}_*} [\Gamma_w] = \frac{\sum_{w \in W_*} \alpha(w) \Gamma_w}{\sum_{w \in W_*} \alpha(w)} \leq \frac{\alpha_{\max} \sum_{w \in W} \Gamma_w}{|W_*| \alpha_{\min}} \leq \frac{\alpha_{\max} \beta_{\max} \cdot O(n \ln n)}{|W_*| \alpha_{\min}} \leq O(\ln n).$$

Thus, by lemma 2.3, the probability that the proposal by  $m_*$  will be accepted is

$$p_1 := \mathbb{E}_{w \sim \mathcal{W}_*} \left[ \frac{\beta_*}{\beta_* + \Gamma_w} \right] \geq \frac{\beta_*}{\beta_* + \mathbb{E}_{w \sim \mathcal{W}_*} [\Gamma_w]} \geq \Omega(1/\ln n).$$

where the first inequality is due to Jensen's inequality.

2. If  $m_*$  proposes to  $w$  and is accepted, then the subsequent rejection chain can either end at the last woman without proposals,  $w_{\text{last}}$ , or cycles back to  $w$  who this time rejects  $m_*$ . Notice that for each subsequent proposal, the ratio between the probability that it goes to  $w_{\text{last}}$  (in which case the process will be terminated) and the probability that it returns to  $w$  is at most  $\alpha_{\max} : \alpha_{\min}$  (and possibly less if the proposing man has already proposed to  $w$ ). Hence, the probability that the chain ends at the last women  $w_{\text{last}}$  is bounded below by

$$p_2 := \frac{\alpha_{\min}}{\alpha_{\max} + \alpha_{\min}} \geq \Omega(1).$$

Note that this is ignoring the chance that a new proposal by  $w$  is rejected, but it still suffices for a lower bound.

3. The probability that  $m_*$  makes more than  $K \ln^2 n$  proposals is thus bounded above by

$$(1 - p_1 p_2)^{K \ln^2 n} \leq \exp(-p_1 p_2 K \ln^2 n) = \exp(-\Omega(K \ln n)) \leq n^{-C}$$

as long as we choose  $K = \Omega(C)$  large enough.  $\blacktriangleleft$

► **Corollary 3.6.** *For any constant  $C \geq 1$ , with probability  $1 - 1/n^C$ , the maximum number of proposals made by any man in DA is  $O(C \ln^2 n)$ .*

**Proof.** By 3.4 and the upper bound for coupon collector (see the full version of this paper), the total number of proposals made in DA is  $O(Cn \ln n)$  with probability  $1 - 1/n^C$ . In particular, if we consider any  $m_*$  and let all other agents propose, this will be true. Recall that by lemma 2.1, DA is independent of the order in which men are chosen to propose. Thus, for each man  $m_*$  we can apply lemma 3.5 to get that, with probability  $1 - 1/n^{C+1}$ ,  $m_*$  makes fewer than  $O((C+1) \ln^2 n) = O(C \ln^2 n)$  proposals. Taking a union bound over the  $n$  men gets the desired result.  $\blacktriangleleft$

► **Remark 3.7.** Both of the above results hold for deferred acceptance with re-proposals as well as deferred acceptance. Indeed, even with re-proposals, deferred acceptance will be independent of the order of proposals (as re-proposals are ignored by the women). Moreover, the logic required to prove points 1. and 2. of the proof of lemma 3.5 is only easier to prove when men sample over all of  $W$  as opposed to just the set  $W_*$ .

The above result is enough to show that proposition 3.2 holds for DA as well for the coupon collector, because repeated proposals are at most a  $O(\ln^2 n/n) = o(1)$  fraction of total proposals in deferred acceptance with re-proposals. We defer the proof to the full version.

► **Theorem 3.8.** *Let  $S$  be the total number of proposals made in DA with tiers of women  $\epsilon, \alpha$ , and arbitrary constant tiers on the men. We have*

$$\mathbb{E}[S] = (1 - O(\ln^2 n/n)) \mathbb{E}[T] = (1 \pm O(1/\ln n)) \frac{\epsilon \cdot \alpha}{\alpha_{\min}} n \ln n.$$

## 4 Rank Achieved by the Men

Up until this point, our arguments have only crudely considered the preferences women have for men. Due to the asymmetry across the different tiers, this means we cannot yet calculate the expected rank men get.

Consider a man  $m$  in tier  $j$ . Our main goal is to prove that the rank of  $m$  is inversely proportional to  $\beta_j$ . As in 3.5, the core tool of our proof will be the fact that deferred acceptance is independent of execution order (by 2.1), and thus we can wait until all other men have finished proposing and found a match before letting  $m$  propose. Once this is done, the major ideas are

1. Suppose  $m$  has public score 1, and define

$$p = \mathbb{E}_{w \sim \mathcal{W}} [\mathbb{P}[w \text{ accepts a proposal from } m]].$$

Note that, if  $m$  were able to propose to a woman independently multiple times, the number of proposals until  $m$  gets his first acceptance would be distributed exactly according to  $\text{Geo}(p)$ , and the expected value would be  $1/p$ . We show that (because men make much less than  $n$  proposals) the difference due to re-proposals is not large.

2. Because  $m$  is the last man to propose, most women have already seen many proposals and arrived at a decent match. When  $m$  gets his first acceptance, he should thus be likely to stay where he is. We show that, while the probability of  $m$  proposing to more women is non-negligible, it still contributes only  $O(1)$  in expectation. So  $m$ 's expected rank is  $1/p$  up to lower-order terms.
3. Another consequence of a woman  $w$  receiving a large number of proposals is the following:

$$\begin{aligned} & \mathbb{P}[w \text{ accepts a proposal from } m' \text{ with weight } \beta] \\ & \approx \beta \cdot \mathbb{P}[w \text{ accepts a proposal from } m \text{ with weight } 1]. \end{aligned}$$

simply by 2.3 and the fact that  $\beta/(\beta + \Gamma_w) \approx \beta \cdot 1/(1 + \Gamma_w)$  for  $\Gamma_w$  (the sum of public scores of men who proposed to  $w$ ) large. Thus, if  $m$  had public score  $\beta$ , the effective value of  $p$  would be approximately  $\beta p$ , and the expected rank of  $m$  would become approximately  $1/(\beta p)$ . In other words, while we are not able to calculate  $p$  directly, we show that  $p$  scales properly with  $m$ 's score.

4. Finally, we prove that the above holds for most sequences of proposals of men before  $m$ , and thus holds in expectation over the entire execution of DA. Note that the distribution of proposals before  $m$  changes slightly depending on which tier  $m$  is chosen from, but in a large market, we do not expect this to make a big difference.

The biggest difference between the above proof sketch and its implementation is that we focus on *two* men proposing at the end of DA. This serves to address point 4 above – we are able to show that, for the vast majority of sequences of proposals before the last two men, their expected ranks are proportional to the ratio of their scores. Thus, this ratio holds in expectation over all of DA. Focusing on two men also allows us to bound the *correlation* between the two men's ranks, which is crucial for our concentration results.

In our proof, we also formalize what it means for all men other than two to propose, with the notion of a “partial matching state”. Moreover, we give the term *smooth* to those states in which the proof sketch above goes through. Most crucially, in smooth matching states, “most women have received a lot of proposals”, so that the reasoning in points 2 and 3 are valid. Additionally, to address certain technicalities (such as being able to bound the magnitude of the expected number of proposals) we define smooth matching states to not have too many proposals in total.

#### 4.1 Smooth matching states

► **Definition 4.1.** *Given a set of men  $L$ , we define the partial matching state excluding  $L$ , denoted  $\mu_{-L}$ , as follows: Run DA with men in  $M \setminus L$  proposing to  $W$ , and keep track of which proposals were made. More specifically, if  $\mu$  is the (partial) matching resulting from running DA with a set of men  $M \setminus L$  and set of women  $W$ , and  $P = \{(m_{i_\ell}, w_{j_\ell})\}_\ell$  is the set of all tuples  $(m_i, w_j)$  where  $m_i$  proposed to  $w_j$  during this process, then  $\mu_{-L} = (\mu, P)$ .*

In a random matching market, we consider this state as a random variable. In a tiered random matching market, to specify this random variable, it suffices to give a multiset of tiers which the men in  $L$  belong to. For a fixed  $\mu_{-L}$ , denote by  $\Gamma_w$  the total sum of weights which woman  $w$  received in  $P$ .

Note that the state  $\mu_{-L}$  keeps track of which proposals have been made (in addition to which current matches are formed) before the men in  $L$  propose.

► **Definition 4.2.** We call a partial matching state  $\mu_{-L}$  smooth if the following hold for some constants  $C_1, C_2, C_3 > 0$ :

1. At most  $C_1 n \ln n$  proposals were made to women overall.
2. At most  $n^{1-C_2}$  women have received fewer than  $C_3 \ln n$  proposals.

The constants  $C_1, C_2, C_3$  in the above depend on the tier structure, and can simply be chosen such that the following proposition holds. Our arguments will go through if smoothness holds with respect to any  $C_1, C_2, C_3$  which are held constant as  $n \rightarrow \infty$ .

► **Proposition 4.3.** Let  $L = \{m_1, m_2\}$  be any pair of men. After running deferred acceptance,  $\mu_{-L}$  is smooth with probability  $1 - n^{-\Omega(1)}$ .

Once we know that  $\mu_{-L}$  is smooth, our two main tasks are to show that men's ranks scale inverse-proportionally to their score, and that the ranks of different men do not correlate too highly. These are the main technical novelties of the paper. The exact details are given in the full version.

► **Proposition 4.4.** Suppose  $\mu_{-L}$  is smooth, and let  $r_1$  and  $r_2$  be the ranks of  $m_1$  and  $m_2$  after running DA with  $m_1$  and  $m_2$  starting from  $\mu_{-L}$ . We have

$$\mathbb{E}_L[r_1] = (1 \pm O(1/\ln n)) \frac{\beta_2}{\beta_1} \mathbb{E}_L[r_2].$$

where we use  $\mathbb{E}_L[\cdot]$  to denote taking an expectation over the random process of  $m_1, m_2$  proposing in DA after starting from state  $\mu_{-L}$ .

► **Proposition 4.5.** Suppose  $\mu_{-L}$  is smooth, and let  $r_1$  and  $r_2$  be the ranks of  $m_1$  and  $m_2$  after running DA with  $m_1$  and  $m_2$  starting from  $\mu_{-L}$ . Then we have  $\text{Cov}(r_i, r_j) = O(\ln^{3/2} n)$ .

## 4.2 Expected rank of the men

In this subsection, we show that overall, expected rank scales proportionally to fitness (in addition to under smooth matching states). This allows us to compute the expected rank of the men. The proofs (deferred to the full version) follow by carefully keeping track of the (limited) effect of non-smooth matching states on the expectation.

► **Proposition 4.6.** Let  $r_i$  and  $r_j$  denote the rank of a man in tiers  $i$  and  $j$ . Then we have

$$\mathbb{E}[r_i] = (1 \pm O(1/\ln n)) \frac{\beta_j}{\beta_i} \mathbb{E}[r_j].$$

► **Theorem 4.7.** Let  $\beta^{-1}$  denote the vector  $(1/\beta_i)_i$ . For each tier  $j$ , the rank  $r_j$  of men in tier  $j$  has expectation

$$\mathbb{E}[r_j] = (1 \pm O(1/\ln n)) \frac{\mathbb{E}[S]}{(n\delta \cdot \beta)\beta_j} = (1 \pm O(1/\ln n)) \frac{\epsilon \cdot \alpha}{\alpha_{\min}} \cdot \frac{1}{(\delta \cdot \beta^{-1})} \cdot \frac{\ln n}{\beta_j}.$$

Finally, we also use our results on the covariance of men's ranks to prove concentration. We defer the proof to the full version. At a high level, the proof follows simply because the weak correlation implied by 4.5 means that the variance of the average of the ranks is lower-order (compared to its expectation), so Chebyshev's inequality can be used.

► **Theorem 4.8.** *For any tier  $j$ , let  $\bar{R}_j^M = (\delta_j n)^{-1} \sum_m r_m$  denote the average rank of men in tier  $j$ . Then, for any  $\epsilon > 0$ ,*

$$\bar{R}_j^M = (1 \pm \epsilon) \frac{\epsilon \cdot \alpha}{\alpha_{\min}} \cdot \frac{1}{(\delta \cdot \beta^{-1})} \cdot \frac{\ln n}{\beta_j}$$

*with probability approaching 1 as  $n \rightarrow \infty$ .*

## 5 Expected rank of the women and the distribution of match types

### 5.1 Expected rank of women

We saw in Section 4.2 that men achieve ranks proportional to the inverse of their public scores. In this section, we turn to the women.

To study the rank the women achieve, we need to reason about the number of proposals women receive on average. By theorem 4.8, we expect that for each tier  $j$  of men, the  $\delta_j n$  men make a total number of proposals approximately

$$\frac{\delta_j \beta_j^{-1}}{\delta \cdot \beta^{-1}} \cdot \frac{\alpha \cdot \epsilon}{\alpha_{\min}} n \ln n.$$

Each of these proposals goes to a woman in tier  $i$  with probability  $\pi_i = \alpha_i / (n \epsilon \cdot \alpha)$ , so we expect such a woman to receive approximately  $(\delta_j \beta_j^{-1}) / (\delta \cdot \beta^{-1}) \cdot (\alpha_i / \alpha_{\min}) \ln n$  proposals from men in tier  $j$ . Each of these men has public score  $\beta_j$ , so we expect  $\Gamma_w$ , the total sum of public scores of men proposing to  $w$ , to be roughly

$$\Gamma_w \approx \sum_j \beta_j \frac{\delta_j \beta_j^{-1}}{\delta \cdot \beta^{-1}} \cdot \frac{\alpha_i}{\alpha_{\min}} \ln n = \frac{\alpha_i \ln n}{\alpha_{\min} (\delta \cdot \beta^{-1})}.$$

It is not immediately clear how the above value of  $\Gamma_w$  should translate to the *rank* that  $w$  gets. Unlike in the case where men are uniform, we cannot simply divide  $n$  by the number of proposals which  $w$  receives.

Indeed, suppose a woman  $w$  receives exactly the total sum of weight  $\Gamma_w$  predicted above. What should her rank be? This is essentially the following: across all tiers of  $\delta_j n$  men each, how many do we expect to beat her best proposal so far? The probability that  $w$  ranks a man  $m$  higher than her match, when viewed according to 2.3, is a function only of the weight  $\beta(m)$  of  $m$  and the weight of proposals  $\Gamma_w$  which  $w$  received. Specifically, this probability is  $\beta(m) / (\beta(m) + \Gamma_w) \approx \beta_j / \Gamma_w$ . Summing this across all the men, we get

$$\mathbb{E}[r_w] \approx \sum_m \frac{\beta(m)}{\beta(m) + \Gamma_w} \approx \frac{n \delta \cdot \beta}{\Gamma_w} \approx (\delta \cdot \beta) (\delta \cdot \beta^{-1}) \frac{\alpha_{\min}}{\alpha_i} \cdot \frac{n}{\ln n}.$$

Note that this ignores the fact that a woman will never rank  $m$  higher than her match if that  $m$  already proposed to her during DA. But since  $w$  only likely receives  $\ln n \ll n / \ln n$  proposals, the difference is not noticeable.

It turns out that, with a detailed probabilistic analysis, the above proof sketch goes through. The details are given in the full version of this paper online.



► **Theorem 5.1.** Let  $\bar{R}_i^W = (\epsilon_i n)^{-1} \sum_{w \in T_i} r_w$  denote the average rank of women in tier  $i$ . For all  $\epsilon > 0$ , we have

$$\bar{R}_i^W = (1 \pm \epsilon)(\delta \cdot \beta)(\delta \cdot \beta^{-1}) \frac{\alpha_{\min}}{\alpha_i} \frac{n}{\ln n}$$

with probability approaching 1 as  $n \rightarrow \infty$ .

## 5.2 The distribution of match types

Fix a woman  $w$  in tier  $i$ . We now study the probability that  $w$  is matched to a man from some tier  $j$ . In the previous section, we argued that with high probability  $w$  receives approximately a total of

$$\frac{\delta_j \beta_j^{-1}}{\delta \cdot \beta^{-1}} \cdot \frac{\alpha \cdot \epsilon}{\alpha_{\min}} n \ln n$$

proposals from men in tier  $j$ . Thus, the contribution to  $\Gamma_w$  (the total weight of proposals  $w$  received) from men in tier  $j$  is

$$\Gamma_{j \rightarrow w} \approx \frac{\delta_j}{\delta \cdot \beta^{-1}} \cdot \frac{\alpha \cdot \epsilon}{\alpha_{\min}} n \ln n \approx \delta_j \Gamma_w.$$

Moreover, it turns out that, with high probability, the above holds up to  $(1 \pm \epsilon)$  for all tiers  $j$  simultaneously. Regardless of the order in which  $w$  saw proposals, the probability that  $w$ 's favorite proposal came from a man in tier  $j$  is  $\Gamma_{j \rightarrow w} / \Gamma_w$ . Thus, this probability is approximately  $\delta_j$ . See the full version for a formal implement of the proof.

► **Theorem 5.2.** Consider an arbitrary tier  $i$  of women and  $j$  of men. For all  $\epsilon > 0$ , there is an  $n$  large enough such that the probability that a woman in tier  $i$  matches to a man in tier  $j$  is  $(1 \pm \epsilon)\delta_j$ .

## 6 Summary

The model and findings in this paper contribute to the understanding of random stable matching markets. Indeed, the results quantify the effect of competition that arises from heterogeneous quality in agents, specifically, when the agents fall into different constant-factor tiers of quality. Novel technical tools are developed in order to reason about the proposal dynamics of deferred acceptance.

Relaxing some of the modeling assumptions raises interesting questions that cannot be trivially answered. This includes having non-constant (size, or public score) tiers, personalized private scores which give agents different distributions of preferences, and imbalance in the number of agents on each side of the market. Moreover, it is natural to ask when one should expect the matching to be sorted, i.e., higher tiers will be more likely to match with higher tiers (e.g., [12] demonstrates the presence of sorting in dating markets).

---

## References

- 1 Itai Ashlagi, Mark Braverman, and Avinatan Hassidim. Stability in large matching markets with complementarities. *Operations Research*, 62(4):713–732, 2014.
- 2 Itai Ashlagi, Mark Braverman, Yash Kanoria, and Peng Shi. Communication requirements and informative signaling in matching markets. In *Proceedings of the 2017 ACM Conference on Economics and Computation*, EC '17, page 263, New York, NY, USA, 2017. Association for Computing Machinery. doi:10.1145/3033274.3084093.

- 3 Itai Ashlagi, Yash Kanoria, and Jacob D Leshno. Unbalanced random matching markets: The stark effect of competition. *Journal of Political Economy*, 125(1):69–98, 2017.
- 4 Hedyeh Beyhaghi, Daniela Sabán, and Éva Tardos. Effect of selfish choices in deferred acceptance with short lists. *CoRR*, abs/1701.00849, 2017. [arXiv:1701.00849](#).
- 5 Robert King Brayton. On the asymptotic behavior of the number of trials necessary to complete a set with random selection. *Journal of Mathematical Analysis and Applications*, 7(1):31–61, 1963.
- 6 Linda Cai and Clayton Thomas. The short-side advantage in random matching markets. *arXiv preprint arXiv:1910.04406*, 2019.
- 7 Peter Coles and Ran Shorrer. Optimal truncation in matching markets. *Games and Economic Behavior*, 87:591–615, 2014.
- 8 Aristides Doulas and Vassilis Papanicolaou. The coupon collector’s problem revisited: Asymptotics of the variance. *Advances in Applied Probability - ADVAN APPL PROBAB*, 44, March 2012. [doi:10.1239/aap/1331216649](#).
- 9 David Gale and Lloyd S Shapley. College admissions and the stability of marriage. *The American Mathematical Monthly*, 69(1):9–15, 1962.
- 10 Hugo Gimbert, Claire Mathieu, and Simon Mauras. Two-sided matching markets with correlated random preferences have few stable pairs. *arXiv preprint arXiv:1904.03890*, 2019.
- 11 Yannai A. Gonczarowski. Manipulation of stable matchings using minimal blacklists. In *Proceedings of the Fifteenth ACM Conference on Economics and Computation*, EC ’14, page 449, New York, NY, USA, 2014. Association for Computing Machinery. [doi:10.1145/2600057.2602840](#).
- 12 Gunter J Hitsch, Ali Hortaçsu, and Dan Ariely. Matching and sorting in online dating. *American Economic Review*, 100(1):130–63, 2010.
- 13 Nicole Immorlica and Mohammad Mahdian. Incentives in large random two-sided markets. *ACM Transactions on Economics and Computation (TEAC)*, 3(3):1–25, 2015.
- 14 Yash Kanoria, Seungki Min, and Pengyu Qian. Which random matching markets exhibit a stark effect of competition? *arXiv preprint arXiv:2006.14653*, 2020.
- 15 D. E. Knuth. *Mariages Stable*. Université de Montréal Press, 1976. Translated as “Stable Marriage and Its Relation to Other Combinatorial Problems, CRM Proceedings and Lecture Notes, 1997.
- 16 Donald E Knuth, Rajeev Motwani, and Boris Pittel. Stable husbands. *Random Structures & Algorithms*, 1(1):1–14, 1990.
- 17 Fuhito Kojima and Parag A Pathak. Incentives and stability in large two-sided matching markets. *American Economic Review*, 99(3):608–27, 2009.
- 18 SangMok Lee. Incentive Compatibility of Large Centralized Matching Markets. *The Review of Economic Studies*, 84(1):444–463, September 2016. [doi:10.1093/restud/rdw041](#).
- 19 David G McVitie and Leslie B Wilson. Stable marriage assignment for unequal sets. *BIT Numerical Mathematics*, 10(3):295–309, 1970.
- 20 Boris Pittel. The average number of stable matchings. *SIAM Journal on Discrete Mathematics*, 2(4):530–549, 1989.
- 21 Boris Pittel. On likely solutions of a stable marriage problem. *The Annals of Applied Probability*, pages 358–401, 1992.
- 22 Boris Pittel. On likely solutions of the stable matching problem with unequal numbers of men and women. *Mathematics of Operations Research*, 44(1):122–146, 2019.
- 23 LB Wilson. An analysis of the stable marriage assignment algorithm. *BIT Numerical Mathematics*, 12(4):569–575, 1972.

# Black-Box Uselessness: Composing Separations in Cryptography

Geoffroy Couteau 

CNRS, IRIF, Université de Paris, France  
couteau@irif.fr

Pooya Farshim 

Department of Computer Science, University of York, UK  
pooya.farshim@york.ac.uk

Mohammad Mahmoody 

Department of Computer Science, University of Virginia, Charlottesville, VA, USA  
mohammad@virginia.edu

---

## Abstract

Black-box separations have been successfully used to identify the limits of a powerful set of tools in cryptography, namely those of black-box reductions. They allow proving that a large set of techniques are not capable of basing one primitive  $\mathcal{P}$  on another  $\mathcal{Q}$ . Such separations, however, do not say anything about the power of the *combination* of primitives  $\mathcal{Q}_1, \mathcal{Q}_2$  for constructing  $\mathcal{P}$ , even if  $\mathcal{P}$  cannot be based on  $\mathcal{Q}_1$  or  $\mathcal{Q}_2$  alone.

By introducing and formalizing the notion of *black-box uselessness*, we develop a framework that allows us to make such conclusions. At an informal level, we call primitive  $\mathcal{Q}$  black-box useless (BBU) for  $\mathcal{P}$  if  $\mathcal{Q}$  cannot help constructing  $\mathcal{P}$  in a black-box way, even in the presence of another primitive  $\mathcal{Z}$ . This is formalized by saying that  $\mathcal{Q}$  is BBU for  $\mathcal{P}$  if for *any* auxiliary primitive  $\mathcal{Z}$ , whenever there exists a black-box construction of  $\mathcal{P}$  from  $(\mathcal{Q}, \mathcal{Z})$ , then there must already also exist a black-box construction of  $\mathcal{P}$  from  $\mathcal{Z}$  alone. We also formalize various other notions of black-box uselessness, and consider in particular the setting of *efficient* black-box constructions when the number of queries to  $\mathcal{Q}$  is below a threshold.

Impagliazzo and Rudich (STOC'89) initiated the study of black-box separations by separating key agreement from one-way functions. We prove a number of initial results in this direction, which indicate that one-way functions are perhaps also *black-box useless* for key agreement. In particular, we show that OWFs are black-box useless in any construction of key agreement in either of the following settings: (1) the key agreement has perfect correctness and one of the parties calls the OWF a constant number of times; (2) the key agreement consists of a single round of interaction (as in Merkle-type protocols). We conjecture that OWFs are indeed black-box useless for general key agreement.

We also show that certain techniques for proving black-box separations can be lifted to the uselessness regime. In particular, we show that the lower bounds of Canetti, Kalai, and Paneth (TCC'15) as well as Garg, Mahmoody, and Mohammed (Crypto'17 & TCC'17) for assumptions behind indistinguishability obfuscation (IO) can be extended to derive black-box uselessness of a variety of primitives for obtaining (approximately correct) IO. These results follow the so-called “compiling out” technique, which we prove to imply black-box uselessness.

Eventually, we study the complementary landscape of black-box uselessness, namely black-box *helpfulness*. We put forth the conjecture that one-way functions are black-box *helpful* for building collision-resistant hash functions. We define two natural relaxations of this conjecture, and prove that both of these conjectures are implied by a natural conjecture regarding random permutations equipped with a collision finder oracle, as defined by Simon (Eurocrypt'98). This conjecture may also be of interest in other contexts, such as amplification of hardness.

**2012 ACM Subject Classification** Theory of computation → Computational complexity and cryptography

**Keywords and phrases** Black-Box Reductions, Separations, One-Way Functions, Key Agreement



© Geoffroy Couteau, Pooya Farshim, and Mohammad Mahmoody;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 47; pp. 47:1–47:20



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Digital Object Identifier 10.4230/LIPIcs.ITCS.2021.47

Related Version Full Version: <https://eprint.iacr.org/2021/016>

Funding *Geoffroy Couteau*: Supported by ERC Project PREP-CRYPTO (724307).

*Mohammad Mahmoody*: Supported by NSF awards CCF-1910681 and CNS1936799.

## 1 Introduction

### 1.1 Background

Black-box reductions are a central tool in Cryptography. They have helped shape a rich landscape of relations between different cryptographic notions, allowing us to develop a better understanding of their powers and limitations.

Roughly speaking, in a (fully) black-box reduction both the design and analysis of a protocol treat the underlying primitives and adversaries in a black-box way, obviously of their internals. More precisely, we say there is a fully black-box construction of a primitive  $\mathcal{P}$  from a primitive  $\mathcal{Q}$  if there is an efficient construction  $\mathcal{P}^{\mathcal{Q}}$  that for every implementation  $\mathcal{Q}$  of primitive  $\mathcal{Q}$  implements primitive  $\mathcal{P}$ , and further, there is an efficient security reduction  $\mathcal{S}^{\mathcal{Q},A}$  which for every adversary  $A^{\mathcal{P}}$  that breaks  $\mathcal{P}$ , breaks  $\mathcal{Q}$ . This notion originates in the seminal work of Impagliazzo and Rudich [20], and it was later refined by Reingold, Trevisan, and Vadhan [26] as well as Baecher, Brzuska, and Fischlin [3] who proposed a taxonomy of notions of reducibility between cryptographic primitives.

Impagliazzo and Rudich showed how to attack any key-agreement (KA) protocol in the random-oracle (RO) model with only a polynomial number queries to the random oracle.<sup>1</sup> This result is sufficient to rule out fully black-box reductions, since, roughly speaking, the construction is assumed to work for any OWF oracle  $f$ , and in particular for a RO and moreover, the security reduction works for any adversary  $A$ , and in particular for those that do not necessarily run in polynomial time but makes a polynomial number of queries to  $f$ .

Following this work, a successful and long line of research studying separations between different cryptographic primitives followed (e.g., see [2, 4, 9, 13, 16, 17, 28] and references therein). In this work we will revisit these works and ask if and to what extent their results hold in the presence of other primitives.

### 1.2 Black-Box Uselessness

Cryptographic constructions typically rely on multiple, incomparable building blocks. This raises the question if, and to what extent, a black-box separation result proves that some primitive is *useless* for building another even with other primitives. Take, for example, the case of key-agreement (KA) and one-way functions (OWFs). Although OWFs on their own are insufficient to build KA, this leaves the possibility that together with some other primitive  $\mathcal{Z}$  they do imply KA in a black-box way, even if  $\mathcal{Z}$  also does not black-box imply KA.<sup>2</sup> More generally, suppose we have separated primitive  $\mathcal{P}$  from primitives  $\mathcal{Q}_1$  and  $\mathcal{Q}_2$  with respect to black-box reductions. That is, neither  $\mathcal{Q}_1$  nor  $\mathcal{Q}_2$  can be used to build  $\mathcal{P}$ . Does it then necessarily follow that  $\mathcal{P}$  is also black-box separated from  $\mathcal{Q}_1$  and  $\mathcal{Q}_2$  put together? More generally, one may ask:

*Which black-box impossibility results compose?*

<sup>1</sup> More precisely, their attack made  $\mathcal{O}((qm)^3)$  RO queries, where  $q$  is the number of queries of the protocol to the RO and  $m$  is the number of messages exchanged.

<sup>2</sup> Note that constructions of PKE from OWFs plus indistinguishability obfuscation are *non-black-box* [27].

In general, not all black-box impossibility results compose. Indeed, consider a primitive  $\mathcal{P}$  that is set to be the “union” of primitives  $\mathcal{Q}_1$  and  $\mathcal{Q}_2$ , where  $\mathcal{Q}_1$  and  $\mathcal{Q}_2$  are mutually separated (i.e., neither can be based on the other). Then although  $\mathcal{P}$  cannot be based on  $\mathcal{Q}_1$  or  $\mathcal{Q}_2$ , it can be trivially based on their union.<sup>3</sup> This situation is somewhat unsatisfying: due to a joint effort of the cryptographic community in the past three decades, we have at our disposal a large number of black-box separations between pairs of primitives, yet we know essentially nothing about whether this situation changes if we are willing to use several primitives in a black-box construction – and of course, black-box separating subsets of primitives from a target primitive would be tedious. This leaves out the possibility that such separations could be obtained more systematically.

In this work, we seek to devise a more efficient strategy, by identifying conditions under which a primitive  $\mathcal{P}$  is black-box separated from primitive  $\mathcal{Q}$  in a *composable way*. That is, primitive  $\mathcal{Q}$  in conjunction with *any* other primitive  $\mathcal{Z}$  cannot be used to build  $\mathcal{P}$ , unless of course  $\mathcal{P}$  can be built using  $\mathcal{Z}$  alone. Our starting point is the following notion of *black-box uselessness*:<sup>4</sup>

► **Definition 1** (Black-box uselessness, informal). *A primitive  $\mathcal{Q}$  is black-box useless (BBU) for primitive  $\mathcal{P}$  if for any auxiliary primitive  $\mathcal{Z}$ , whenever there is a black-box construction of  $\mathcal{P}$  from  $(\mathcal{Q}, \mathcal{Z})$  there also exist a black-box construction of  $\mathcal{P}$  from the auxiliary primitive  $\mathcal{Z}$  alone.*

A *composition theorem* for black-box uselessness immediately follows: if  $\mathcal{Q}_1$  is BBU for  $\mathcal{P}$  and  $\mathcal{Q}_2$  is BBU for  $\mathcal{P}$  then  $\mathcal{Q}_1$  and  $\mathcal{Q}_2$  put together are BBU for  $\mathcal{P}$ . Indeed, for any  $\mathcal{Z}$ , if  $(\mathcal{Q}_1, \mathcal{Q}_2, \mathcal{Z}) = (\mathcal{Q}_1, (\mathcal{Q}_2, \mathcal{Z}))$  black-box implies  $\mathcal{P}$ , then  $(\mathcal{Q}_2, \mathcal{Z})$  must black-box imply  $\mathcal{P}$  (by the black-box uselessness of  $\mathcal{Q}_1$ ). This in turn implies, by the black-box uselessness of  $\mathcal{Q}_2$ , that  $\mathcal{Z}$  alone black-box imply  $\mathcal{P}$ .

► **Remark 2.** A black-box uselessness result of  $\mathcal{Q}$  for  $\mathcal{P}$  implies in particular that  $\mathcal{Q}$  does not black-box imply  $\mathcal{P}$ , as long as  $\mathcal{P}$  is not a (trivial) primitive that exists unconditionally. Indeed, by the black-box uselessness of  $\mathcal{Q}$ , if  $\mathcal{Q}$  black-box implies  $\mathcal{P}$ , then taking  $\mathcal{Z}$  as the “empty primitive” we get that  $\mathcal{P}$  exists unconditionally in the plain model. For instance, although one-way functions are black-box useless for the one-time pad (since the latter exists unconditionally in the presence of any auxiliary oracle), one-way functions black-box imply the one-time pad (for essentially the same reason).

### 1.3 One-Way Functions and Key Agreement

Perhaps one of the most fundamental questions regarding black-box uselessness is to understand whether or not one-way functions are black-box useless for key agreement. We start with an observation on a natural approach for building key-agreement from one-way functions together with other primitives.

<sup>3</sup> This formal counterexample can be converted into more “natural” ones. Take identity-based encryption (IBE) with *compact* user keys, whose lengths are independent of the length of user identities. One can use a standard IBE and a collision-resistance hash function (CRHF) to build a compact IBE (by simply hashing the identities). Yet, compact IBE cannot be based on CRHFs in a black-box way (since PKE cannot be based on ROs). Furthermore, compact IBE cannot be based on standard IBE, since compact IBE implies CRHF (the keys must be collision-free for otherwise the compact IBE can be broken by finding a collision among keys) and standard IBE does not imply CRHFs [21].

<sup>4</sup> The terminology “a primitive  $X$  is useless for black-box constructions of a primitive  $Y$ ” has been used sometimes in the literature, e.g. in [25], to mean that  $Y$  does not black-box reduce to  $X$ . Our notion of black-box uselessness should not be confused with this terminology, which only refers to conventional black-box separations.

► **Remark 3.** When looking for candidate primitives that, combined with one-way functions, imply key-agreement, the notion of indistinguishability obfuscation (iO, [27]) might be the first that comes to mind to a reader familiar with the literature on cryptography. As we mentioned, however, the construction of PKE from iO+OWFs in [27] is not black-box. Furthermore, one can observe that this is actually unavoidable: the seminal work of Impagliazzo and Rudich [20], which showed that there is no black-box construction of key agreement from one-way function, actually already shows that there is no black-box construction of key agreement from one-way functions *together with iO*. This is because the result of [20] shows that, relative to a random oracle *and a PSPACE oracle*, there is no key-agreement, yet there are one-way functions. However, relative to these oracles, there is also a perfectly-seure deterministic iO scheme: on input a circuit of a given size, the obfuscation scheme uses the PSPACE oracle to efficiently compute the lexicographically-first functionally equivalent circuit of the same size. This simple observation implies in particular that there is no black-box construction of key-agreement from iO and OWFs.

In this work, we provide a partial answer to the question of whether or not one-way functions are black-box useless for key agreement, by showing that one-way functions are black-box useless in any construction of key agreement satisfying certain restrictions. To describe our result, it is instructive to start with the separation of perfectly correct KA from OWFs by Brakerski, Katz, Segev, and Yerukhimovich [9].

### BKSY11 in a nutshell

Given a perfectly correct, two-party KA protocol in the RO model, where Alice and Bob place at most  $q$  queries to the RO, consider an attacker Eve that proceeds as follows. Given a transcript  $T$  of the protocol, Eve samples coins  $r'_A$  and a random oracle  $RO'$  for Alice that are consistent with  $T$ . Eve then runs Alice on  $r'_A$  and  $RO'$  and records all oracle queries and the resulting key. It then places all these queries to the real RO to obtain an assignment (a list of query-answer pairs)  $L$ . Eve repeats this process, appending to  $L$  and storing keys, while ensuring that the sampling of  $RO'$  is consistent with  $L$  computed so far. Now, if in a sampled run of Alice, there are no queries of Alice in common with those of Bob outside  $L$ , by perfect correctness, the correct key will be computed. Otherwise, a query of Bob will be discovered. Since Bob has at most  $q$  queries, if Eve executes this procedure  $2q + 1$  times, in at least  $q$  of the runs no intersection queries will be discovered, and in these runs the correct key will be computed. Thus taking a majority over the set of keys computed, Eve obtains the key with probability one.

### Upgrading to BBU

In order to convert this proof into a black-box uselessness result, we make a case distinction based on whether or not one can “lift” the sampling procedure to a  $Z$ -relativized world for any oracle  $Z$ . Given a construction  $KA^{F,Z}$  of KA from a OWF  $F$  and  $Z$ , if the sampling procedure can be successfully carried out in the presence of  $Z$ , then we could efficiently break any perfectly correct KA protocol in the presence of  $Z$  only (since the attacker computes the correct key with probability one).

Now suppose at some iteration Eve is no longer able to find coins and a random oracle consistent with the transcript while making polynomially many queries to  $Z$ . We claim that in this case we can construct a *weak* one-way function. Indeed, consider the function that runs the above attack procedure up to but excluding the stage where Eve fails. Thus, this function starts by running the protocol, then uses the sampler for the first iteration (if



sampling was not already impossible at this stage) to obtain an assignment, and continues with another round of sampling, until it arrives at the stage where sampling is no longer possible. The transcript of the protocol at this stage together with the list of assignments so far constitutes the challenge for which there is no inverter. From a weak one-way function, a full-fledged one-way function follows by the result of [29], as this construction is black-box and hence relativizes with respect to  $Z$ .

We may now use the one-way function obtained from  $Z$  in place of the original one-way function  $F$  to remove reliance on the  $F$  all together. Thus, we obtain a KA protocol which relies only on  $Z$ , as required. We emphasize that this proof only shows the uselessness of OWFs for (perfect) KA and not that of random oracles, since we only obtain a one-way function using  $Z$ .

Note that since we recursively rely on the existence of an inverter, the query complexity (to  $Z$ ) of the samplers can potentially blow up. Indeed, suppose for some  $Z$ , any sampler needs to place  $\mathcal{O}(n^2)$  queries to  $Z$  to invert a function that places  $n$  queries to  $Z$ . After the first iteration, we arrive at the construction of a function that places  $n + \mathcal{O}(n^2) = \mathcal{O}(n^2)$  queries to  $Z$ . Thus, it may well be that a successful inverter at this step needs to place  $\mathcal{O}(n^4)$  queries to  $Z$ . This in particular implies that the recursive argument above can be applied for only a *constant* number of steps. Since we only need to apply the recursive sampling for *either* Alice *or* Bob, we obtain a BBU result as long as either Alice or Bob makes a constant number of queries to the RO (but polynomially many calls to  $Z$ ). We formalize this proof in Section 4, where we point out the other subtleties that arise.

Currently, we are not able to extend the proof to arbitrary protocols where both Alice and Bob make a polynomial number of RO queries. Despite this, we can show that OWFs are black-box useless for building *constant-round*, *imperfect* key agreements when both parties make a constant number of queries to the OWF (and an arbitrary number of queries to the auxiliary oracle), and that OWFs are black-box useless for *one-round* key agreement (without restriction on the queries made by the parties). We defer the details to Section 4.2.

## 1.4 The Compilation Technique

A number of black-box separation results rely on what we here refer to as the *efficient compiling-out* (or simply compilation) paradigm [1, 7, 10, 12, 13, 15, 16]. At a high-level, here given a construction  $G_1^P$  one compiles out the primitive  $P$  via an (oracle-free) simulator  $\text{Sim}$  which simulates  $P$  for the construction in a consistent way and without affecting security. The result is a new construction  $G_2$  that no longer uses  $P$ . This proof technique is closely related to black-box uselessness, and as we will see can often be turned into a black-box uselessness result with minor modifications. This in turn highlights the advantage of separation results that are achieved using this technique.

In order to show how this can be done, we briefly discuss this in the context of the work of Canetti, Kalai, Paneth [10], who showed that obfuscation cannot be based on random oracles.<sup>5</sup>

<sup>5</sup> The work of [10] dealt with *virtual-black-box* obfuscation, but as it turned out [8, 22, 23], their proof could also be applied to the case of indistinguishability obfuscation.



**CKP and black-box uselessness of random oracles for indistinguishability obfuscation**

Consider an obfuscator  $\text{Obf}^{\text{RO}}$  that makes use of a random oracle RO. On input a circuit without random oracle gates<sup>6</sup> the obfuscation algorithm outputs a circuit  $C^{\text{RO}}$ , which may also call the random oracle RO. CKP compile out the random oracle RO from any such construction as follows. First they convert  $\text{Obf}^{\text{RO}}$  to an obfuscator in the plain model by simulating the RO for the obfuscator via lazy sampling. Next to ensure that the oracle expected by an obfuscated circuit  $C^{\text{RO}}$  is consistent with the oracle simulated for obfuscation, CKP execute  $C$  on multiple randomly chosen inputs at obfuscation phase while simulating the random oracle consistently, record all query made and answers simulated, and return them together with the obfuscated oracle circuit. Leaking this list cannot hurt security as an adversary in the RO model can also compute such a list given an obfuscated circuit. On the other hand, having this list allows the evaluation of  $C^{\text{RO}}$  to be consistent with the obfuscation phase with high probability on a *random* input. (This is why the obtained primitive is only an *approximate-correct* IO.)

As can be seen, this proof Z-relativizes in the sense that the simulation of the random oracle RO and the execution of the obfuscated circuit can be both done in the presence of any other oracle Z. By compiling out the random oracle RO in the presence of Z, we obtain a construction that relies on Z. That is, we obtain that random oracles are black-box uselessness for obfuscation.

A number of other impossibility results follow the compilation paradigm and here we observe that they can be lifted to the black-box uselessness regime. In particular, we go over such observations about results from [10, 13, 15, 16] and show how they can be lifted to black-box uselessness. Indeed, as long as compilation is performed for a constant “number of rounds” and each step can be done efficiently we obtain black-box uselessness.<sup>7</sup> As a result we get that approximate IO cannot be obtained from a black-box *combination* of random oracles, predicate encryption, fully homomorphic encryption and witness encryption.<sup>8</sup>

► **Remark 4.** We note that some previous works (e.g., [10]) have described the result of Impagliazzo and Rudich as “compiling out” the random oracle from any key-agreement protocol. This process, however, differs from the compiling-out technique that we study in this paper in two aspects. First, compilation is inefficient and uses the sampling algorithm. Second, the process is carried out adaptively for multiple rounds. The inefficiency of the sampler translates to obtaining BBU for one-way functions only; adaptivity restricts our final result to protocols where one party makes at most a constant number of RO queries. On a similar note, the recent work of Maji and Wang [25] uses the term “black-box useless” as an alternative to proving (traditional) black-box separations. So, despite similarity in the terms, our notion of uselessness is quite different.

**1.5 The Case of Collision Resistance**

A classic result of Simon [28] separates collision-resistance hash functions (CRHFs) from one-way functions (and indeed one-way permutations). This is done by giving a pair of oracles  $(\pi, \text{Coll}^\pi)$  relative to which one-way permutations (and hence one-way functions) exist,

<sup>6</sup> This restriction only makes the results of CKP stronger.

<sup>7</sup> Lemma 3.25 in [14] shows a similar phenomenon in a context where we completely compile out a sequence of idealized oracles. Here we deal with a setting that an auxiliary oracle remains at the end.

<sup>8</sup> We note that this does not apply in the so-called monolithic model.

but CRHFs don't. Here  $\pi$  implements a random permutation, and  $\text{Coll}^\pi$  is an oracle that takes as input a circuit with  $\pi$ -gates and returns a random collision for it (by first computing the circuit on a random point and then picking a random preimage).

Are one-way functions/permutations also black-box useless for collision resistance? One way to answer this question affirmatively would be to extend Simon's result along the lines of what was done for the separation results above. However, in this case we have an oracle separation result, and it is not clear how to "relativize" the proof. Indeed, we conjecture the opposite: OWFs are black-box *helpful* for building CRHFs. To this end, we would need to show that for *any* one-way function  $F$  there is a primitive  $Z$ , which although on its own is insufficient for building CRHFs, together with  $F$  can be used to build CRHFs. We present two possible approaches for proving this conjecture.

One approach follows the recent result of Holmgren and Lombardi [18], who showed how to obtain CRHFs from exponentially secure *one-way product functions* (OWPFs) in a black-box way. Roughly speaking, a OWP is a tuple of one-way functions  $(F_1, \dots, F_k)$  where any polynomial-time adversary can invert  $(F_1(x_1), \dots, F_k(x_k))$  for random  $x_i$  with probability at most  $\text{negl}(n)/2^n$  (where  $n$  is the security parameter). A good candidate for primitive  $Z$  is thus a random permutation  $\pi$  together with Simon's oracle  $\text{Coll}^\pi$  for it. To get a positive helpfulness result we need to show that for any one-way function  $F$  the pair of functions  $(F, (\pi, \text{Coll}^\pi))$  is product one-way. We intuitively expect this result to hold since  $\pi$  is fully independent of  $F$  and essentially all an adversary can do is to invert the  $F$  and  $(\pi, \text{Coll}^\pi)$  independently. Formalizing this observation requires handling additional technicalities; we refer the reader to the full version for details. We did not manage to prove this conjecture, and leave it as an interesting open problem which might be of independent interest.<sup>9</sup>

A second approach follows the work of Bauer, Farshim, Mazaheri [5], who defined a new idealized model of computation, known as the backdoored random oracle (BRO) model, whereby an adversary can obtain arbitrary leakage of the function table of the RO. Under a communication complexity conjecture related to the set-intersection problem, BFM show that two independent BROs can be used to build a CRHF by simply xoring their outputs. The leakage oracle defined by BFM is sufficiently powerful to allow implementing Simon's collision-finding oracle. As a result, although a single BRO as an idealized primitive is black-box separated from CRHFs, conjecturally it is not black-box useless for building CRHFs.

## Open problems

The central open problem left by our work is that of black-box uselessness of OWFs for arbitrary key agreement protocols. Given our BBU results for special classes of KA protocols, this conjecture may well be within reach. On the other hand, a straightforward generalization seems to require a refined sampler technique with low *adaptivity*. At the moment, however, it seems challenging to reduce the adaptivities of known samplers [4, 9, 20, 24]. Besides OWFs, whether or not CRHFs, or for that matter random oracles, are black-box useless for key agreement remains open. More generally, black-box separation results can be revisited from the point of view of uselessness. In particular, it would be interesting to consider extensions of the recent *monolithic* models to the BBU setting, as these capture certain well-known non-black-box techniques in cryptography.

<sup>9</sup> It might be helpful to consider weaker versions of this problem. For example, given an  $\varepsilon$ -secure one-way permutation and a random oracle, can an attacker invert both simultaneously with probability better than  $\text{negl}(n)/2^n$ ?

## 2 Preliminaries

### Notation

PPT stands for probabilistic polynomial time. An oracle-aided PPT machine/circuit  $A^O$  is a PPT machine/circuit with oracle access/gates, such that for any oracle  $O$  machine/circuit  $A^O$  runs in probabilistic polynomial time, where we count each call to the oracle as one step. For any  $n \in \mathbb{N}$ , we let  $[n]$  denote the set  $\{1, \dots, n\}$ . Given an interactive protocol between two parties Alice and Bob with access to an oracle  $O$ , we let  $\langle \text{Alice}^O, \text{Bob}^O \rangle$  denote the distribution of triples  $(T, y_A, y_B)$  over the randomness of the parties and the oracle, where  $T$  denotes the transcript of the protocol, and  $y_A$  (resp.,  $y_B$ ) denotes the output of Alice (resp., Bob). We also use the same notation when  $O$  comes from a *distribution* over the oracles, in which case the probability distribution of the outputs are also over the randomness of the oracle.

### 2.1 Black-Box Reductions

The definitions and notions in this section mostly follow those in [26]. Throughout, we use calligraphic letters such as  $\mathcal{P}$  or  $\mathcal{KA}$  for a cryptographic primitive, sans-serif letters (for example  $P$  or  $KA$ ) for specific implementations,  $S$  for the security reduction/proof and  $P$  for a “generic” implementation. We denote an auxiliary oracle by  $Z$  and an adversary by  $A$ .

► **Definition 5** (Cryptographic primitive). *A cryptographic primitive  $\mathcal{P}$  is a pair  $(F_{\mathcal{P}}, R_{\mathcal{P}})$ , where  $F_{\mathcal{P}}$  is the set of functions implementing  $\mathcal{P}$ , and  $R_{\mathcal{P}}$  is a relation. For each  $P \in F_{\mathcal{P}}$ , the relation  $(P, A) \in R_{\mathcal{P}}$  means that the adversary  $A$  breaks the implementation  $P$  (according to  $\mathcal{P}$ ). It is required that at least one function  $P \in \mathcal{P}$  is computable by a PPT algorithm.*

► **Definition 6** (Fully black-box reduction). *A fully black-box reduction of a primitive  $\mathcal{P}$  to another primitive  $\mathcal{Q}$  is a pair  $(P, S)$  of oracle-aided PPT machine such that for any implementation  $Q \in \mathcal{Q}$  the following two conditions hold.*

- **Implementation reduction:**  $P^Q$  implements  $\mathcal{P}$ , that is,  $P^Q \in F_{\mathcal{P}}$ .
- **Security reduction:** For any function (adversary)  $A$  that  $\mathcal{P}$ -breaks  $P^Q \in F_{\mathcal{P}}$ , i.e.,  $(P^Q, A) \in R_{\mathcal{P}}$ , it holds that  $S^{Q,A}$   $\mathcal{Q}$ -breaks  $Q$ , i.e.,  $(Q, S^{Q,A}) \in R_{\mathcal{Q}}$ .

When clear from context, we will refer to fully black-box reductions simply as black-box reductions, to the implementation reduction as the *construction*, and to the security reduction as the *security proof*.

► **Definition 7** (Semi and weakly black-box reductions). *We say there is a semi-black-box reduction of a primitive  $\mathcal{P}$  to primitive  $\mathcal{Q}$  if there is an oracle-aided PPT  $P$  such that for any implementation  $Q \in \mathcal{Q}$ ,*

- **Implementation reduction:**  $P^Q \in F_{\mathcal{P}}$ .
- **Security reduction:** If there exists an oracle-aided PPT  $A$  such that  $A^Q$   $\mathcal{P}$ -breaks  $P^Q$ , then there exists an oracle-aided PPT  $S$  such that  $S^Q$   $\mathcal{Q}$ -breaks  $Q$ .

*If the order of the quantifiers for the implementation reduction is switched in the sense that for any implementation  $Q \in \mathcal{Q}$  there is an oracle-aided PPT  $P$  such that the above two conditions hold, then we say there is a  $\forall\exists$ -semi-black-box reduction of  $\mathcal{P}$  to  $\mathcal{Q}$ .*

*Weakly black-box reductions and a  $\forall\exists$  variant thereof are defined analogously with the difference that the security reduction  $S$  is a PPT (instead of an oracle-aided PPT) machine and can no longer call  $Q$ .*

► **Definition 8** (Existence relative to an oracle). *A primitive  $\mathcal{P}$  is said to exist relative to an oracle  $O$  whenever (1) there is an oracle-aided PPT  $P$  such that  $P^O \in F_{\mathcal{P}}$ ; and (2) no oracle-aided PPT machine  $A^O$  can  $\mathcal{P}$ -break  $P^O$ .*

*Non-uniform* variants of security reductions were formalized in [11]. The following definition extends this to non-uniform implementation reductions.

► **Definition 9.** A fully black-box reduction  $(P, S)$  of a primitive  $\mathcal{P}$  from primitive  $\mathcal{Q}$  is said to have a non-uniform implementation if  $P$  additionally takes as input a polynomial-sized non-uniform advice that can also depend on its oracle  $\mathcal{Q}$ . The reduction is said to have a non-uniform security reduction if  $S$  additionally takes as input a polynomial-sized non-uniform advice that can also depend on its oracles  $\mathcal{Q}$  and  $A$ .

## 2.2 Specific Cryptographic Primitives

► **Definition 10** (One-way functions). A one-way function  $F$  relative to an oracle  $\mathcal{O}$  is an oracle-aided PPT machine such that for any PPT adversary  $A$  (modeled as an oracle-aided PPT machine) and any sufficiently large values of the security parameter  $\lambda$ , it holds that

$$\Pr_{x \xleftarrow{\$} \{0,1\}^n} [F^{\mathcal{O}}(A^{\mathcal{O}}(1^\lambda, F^{\mathcal{O}}(x))) = F^{\mathcal{O}}(x)] = \text{negl}(\lambda) .$$

If the above is only required to hold for infinitely many values of  $\lambda \in \mathbb{N}$ , then  $F$  is an infinitely-often one-way function (*io-F*).

► **Definition 11** (Key agreement). An oracle-aided  $\varepsilon$ -key agreement with respect to an oracle  $\mathcal{O}$  is an interactive protocol between two oracle-aided PPT machines Alice and Bob that satisfies the following  $\varepsilon$ -correctness and security properties.

■ ( $\varepsilon$ -correctness) For any  $\lambda \in \mathbb{N}$ ,

$$\Pr[(T, K_A, K_B) \xleftarrow{\$} \langle \text{Alice}^{\mathcal{O}}(1^\lambda), \text{Bob}^{\mathcal{O}}(1^\lambda) \rangle : K_A = K_B] \geq \varepsilon(\lambda) .$$

■ (Security) For any PPT adversary Eve, any polynomial  $p$ , and any sufficiently large  $\lambda$ ,

$$\Pr[(T, K_A, K_B) \xleftarrow{\$} \langle \text{Alice}^{\mathcal{O}}(1^\lambda), \text{Bob}^{\mathcal{O}}(1^\lambda) \rangle, K_E \xleftarrow{\$} \text{Eve}^{\mathcal{O}}(1^\lambda, T) : K_E = K_B] \leq \frac{\varepsilon(\lambda)}{p(\lambda)} .$$

If security is only required to hold for infinitely many values of  $\lambda \in \mathbb{N}$  in the sense that for every polynomial  $p$  and all adversaries, there exists an infinite set of  $\lambda$  for which the adversary's winning probability is below  $\varepsilon(\lambda)/p(\lambda)$ , then the construction is called an infinitely-often key agreement. If the number of queries to  $\mathcal{O}$  is bounded by a constant for either Alice or Bob, then we say that the key agreement is unbalanced with respect to  $\mathcal{O}$ . We say that the key agreement is a perfectly correct when  $\varepsilon(\lambda) \equiv 1$ .

## 3 Defining Black-Box Uselessness

To formally define black-box uselessness, we first formalize joint primitives, to simplify statements about black-box constructions from several primitives:

► **Definition 12** (Joint primitive). Given two primitives  $\mathcal{P} = (F_{\mathcal{P}}, R_{\mathcal{P}})$  and  $\mathcal{Q} = (F_{\mathcal{Q}}, R_{\mathcal{Q}})$  the joint primitive  $(\mathcal{P}, \mathcal{Q}) = (F_{(\mathcal{P}, \mathcal{Q})}, R_{(\mathcal{P}, \mathcal{Q})})$  is defined by  $F_{(\mathcal{P}, \mathcal{Q})} := F_{\mathcal{P}} \times F_{\mathcal{Q}}$  and for each  $G = (P, Q) \in F_{(\mathcal{P}, \mathcal{Q})}$  and any  $A = (A_{\mathcal{P}}, A_{\mathcal{Q}})$ , we define  $(G, A) \in R_{(\mathcal{P}, \mathcal{Q})}$  iff  $(P, A_{\mathcal{P}}) \in R_{\mathcal{P}}$  or  $(Q, A_{\mathcal{Q}}) \in R_{\mathcal{Q}}$ .

We are now ready to formally define what it means for a cryptographic primitive  $\mathcal{Q}$  to be black-box useless for a primitive  $\mathcal{P}$ .

### 3.1 Definition

The following definition is more general than complete black-box uselessness of a primitive  $\mathcal{P}$  for obtaining another primitive  $\mathcal{Q}$ . In particular, the definition states a *set* of primitives  $\mathbb{Z}$  such that  $\mathcal{P}$  is useless for obtaining  $\mathcal{Q}$  in the presence of *any* of the primitives  $\mathcal{R} \in \mathbb{Z}$ .

► **Definition 13** (Black-box uselessness). *Suppose  $\mathbb{Z}$  is a set of primitives. A cryptographic primitive  $\mathcal{Q}$  is resp., fully, semi,  $\forall\exists$ -semi black-box useless for constructing a primitive  $\mathcal{P}$  in the presence of auxiliary primitives  $\mathbb{Z}$ , if the following holds: For every auxiliary primitive  $\mathcal{Z} \in \mathbb{Z}$  whenever there exists a resp., fully, semi,  $\forall\exists$ -semi black-box reduction of  $\mathcal{P}$  to the joint primitive  $(\mathcal{Q}, \mathcal{Z})$  there also exists a resp., fully, semi,  $\forall\exists$ -semi black-box reduction of  $\mathcal{P}$  to  $\mathcal{Z}$  alone. In the special case where  $\mathbb{Z}$  contains all primitives, then we simply say that  $\mathcal{Q}$  is resp., fully, semi,  $\forall\exists$ -semi black-box useless for constructing a primitive  $\mathcal{P}$ .*

► **Remark 14** (Other special cases of Definition 13). Here we point out to two other important special cases of Definition 13 that could be obtained  $\mathbb{Z}$  differently. If  $\mathbb{Z} = \emptyset$ , then black-box uselessness is the same as a traditional black-box separation showing that  $\mathcal{Q}$  cannot be black-box reduced to  $\mathcal{P}$ . In addition, when  $\mathbb{Z}$  contains a specific primitive  $\mathcal{Z}$ , then Definition 13 captures the notion that  $\mathcal{P}$  is useless for building  $\mathcal{Q}$  when we already assume the existence of  $\mathcal{Z}$  as a black-box.

► **Remark 15** (Other variants of Definition 13). By default we consider the three main cases where the source construction and the target construction in Definition 13 both use the same flavor of black-box reduction, and accordingly use the terms fully, semi, or  $\forall\exists$ -semi to describe the corresponding notion of black-box uselessness. However, we can (and will) also consider more general notions of black-box uselessness whereby the source construction and the target construction use different notions of black-box reduction. For example, we will write that  $\mathcal{Q}$  is [semi  $\rightarrow$   $\forall\exists$ -semi] black-box useless for  $\mathcal{P}$  if for every auxiliary primitive  $\mathcal{Z}$ , whenever there exists a semi-black-box reduction of  $\mathcal{P}$  to the joint primitive  $(\mathcal{Q}, \mathcal{Z})$  there is a  $\forall\exists$ -semi-black-box reduction of  $\mathcal{P}$  to  $\mathcal{Z}$  alone.

### 3.2 Composition

Given the definition of black-box uselessness, the following composition theorem follows easily. Here, we use the term black-box uselessness to refer to any fixed flavor of black-box uselessness.

► **Theorem 16.** *Let  $\mathcal{P}$ ,  $\mathcal{Q}$  and  $\mathcal{R}$  be three cryptographic primitives. If  $\mathcal{Q}$  is black-box useless for  $\mathcal{P}$  and  $\mathcal{R}$  is black-box useless for  $\mathcal{P}$  (for the same flavor), then the joint primitive  $(\mathcal{Q}, \mathcal{R})$  is black-box useless for  $\mathcal{P}$  (for the same flavor).*

**Proof.** Let  $\mathcal{Z}$  be an arbitrary auxiliary primitive. If there is a black-box reduction of  $\mathcal{P}$  to the joint primitive  $(\mathcal{Z}, (\mathcal{Q}, \mathcal{R})) = ((\mathcal{Z}, \mathcal{Q}), \mathcal{R})$ , then by the black-box uselessness of  $\mathcal{R}$  for  $\mathcal{P}$ , there is a black-box reduction of  $\mathcal{P}$  to  $(\mathcal{Z}, \mathcal{Q})$  (viewing  $(\mathcal{Z}, \mathcal{Q})$  as an auxiliary primitive). In turn, using the black-box uselessness of  $\mathcal{Q}$  for  $\mathcal{P}$ , we obtain a black-box construction of  $\mathcal{P}$  from  $\mathcal{Z}$  alone. Hence,  $(\mathcal{Q}, \mathcal{R})$  is black-box useless for  $\mathcal{P}$ . ◀

We note that a similar composition theorem can be easily established even when  $\mathcal{Q}$  and  $\mathcal{R}$  do not satisfy the same flavor of black-box uselessness for  $\mathcal{P}$ , in the following sense: if a primitive  $\mathcal{Q}$  is  $[X \rightarrow Y]$  BBU for  $\mathcal{P}$  and a primitive  $\mathcal{R}$  is  $[X' \rightarrow Y']$  BBU for  $\mathcal{P}$ , where  $X, Y, X', Y'$  are flavors of black-box reduction, then  $(\mathcal{Q}, \mathcal{R})$  is  $[X' \rightarrow Y]$  BBU for  $\mathcal{P}$  as long as  $X$  is a stronger flavor than  $Y'$  (e.g.,  $X$  is “fully” and  $Y'$  is “semi”).

### 3.3 Restricted Black-Box Uselessness

In many settings, it can be useful to consider a more general notion of black-box uselessness, which restricts the type of primitive (e.g., only infinitely-often variants) or the type of construction for which black-box uselessness is shown to hold. For readability, we will not define cumbersome formal notations for such variants, but instead will simply state the restriction explicitly when needed.

This generalization is especially useful to study the *efficiency* of black-box reductions. Indeed, black-box separations in cryptography are not limited to only showing their nonexistence. A more concrete treatment would make statements of the form “in any black-box construction of  $\mathcal{P}$  from  $\mathcal{Q}$ , any implementation of  $\mathcal{P}$  must call an implementation of  $\mathcal{Q}$  at least  $q$  times,” or for interactive primitives states that “any black-box construction of  $\mathcal{P}$  from  $\mathcal{Q}$  must have at least  $r$  rounds”. This approach to bounding the efficiency of generic cryptographic construction was initiated in the seminal work of Gennaro, Gertner, Katz, and Trevisan [16] and has subsequently proven very fruitful.

## 4 On the Black-Box Uselessness of OWFs for Key Agreement

In this section, we prove black-box uselessness of OWFs for key agreement for several natural special forms of key agreement protocols. We leave the proof of black-box uselessness of OWFs for general key agreement protocols as an intriguing open question.

### 4.1 Theorem Statement for Perfectly Correct Key Agreement

In this section, we prove the following:

► **Theorem 17** (Black-box uselessness of OWFs for perfect unbalanced KA). *Infinitely-often one-way functions are [semi  $\rightarrow$   $\forall\exists$ -semi] black-box useless for infinitely-often perfect key agreement in any construction which is unbalanced with respect to the io-OWF.*

Before proving Theorem 17, let us breakdown its content. The “dream result” here would be to show that one-way functions are black-box useless for any key agreement. Unfortunately, we do not know how to prove this result. Theorem 17 provides a meaningful step in this direction, but it suffers from three limitations:

1. It only applies to *infinitely-often* one-way functions (though it is incomparable in that it shows black-box uselessness for infinitely-often key agreement, which is a weaker primitive).
2. It only applies to constructions where one of the parties makes a *constant* number of queries to the io-OWF oracle, which we call unbalanced key agreement. Note that the key agreement can still make an arbitrary number of queries to the auxiliary primitive.
3. It only applies to *perfectly correct* key agreement.

The first limitation stems from the fact that our proof of Theorem 17 relies on a case distinction based on the existence of one-way functions: if they exist, we get a construction of key agreement, else we get an attack on the candidate construction. However, this attack requires applying a one-way function inverter to several functions at once. But since a one-way function inverter is only guaranteed to succeed on *infinitely many* security parameters, which need not be equal across the different functions that we need to invert (and in fact could be exponentially far apart), this approach fails. To get around this, we rely on an inverter for an infinitely-often OWF, which gives an inverter which is guaranteed to work for *all* sufficiently large security parameters and we can use to simultaneously invert several functions. This,



however, comes at the cost of getting instead a black-box uselessness result for infinitely-often OWFs (for infinitely-often key agreements). Such technicalities are relatively common in cryptography and stem from the asymptotic nature of primitives.

► **Remark 18.** In general, statements of the form “ $\mathcal{A}$  and  $\mathcal{B}$  black-box imply  $\mathcal{C}$ ” and statements of the form “io- $\mathcal{A}$  and io- $\mathcal{B}$  black-box imply io- $\mathcal{C}$ ”, where io- $\mathcal{X}$  denotes an infinitely-often flavor of a primitive  $\mathcal{X}$ , can be incomparable for the trivial reason that io- $\mathcal{A}$  and io- $\mathcal{B}$  can never be simultaneously secure on the same security parameters. However, this situation does not arise in the setting of black-box uselessness, since the statement “io- $\mathcal{A}$  is BBU for io- $\mathcal{C}$ ” refers to the inexistence of black-box construction of io- $\mathcal{C}$  from io- $\mathcal{A}$  together with any other primitive  $\mathcal{Z}$  – and not only “infinitely-often” types of primitives. In general, it is easy to show that the statement “ $\mathcal{A}$  is BBU for  $\mathcal{C}$ ” is stronger than (i.e., implies) the statement “io- $\mathcal{A}$  is BBU for io- $\mathcal{C}$ ” for all notions of black-box uselessness.

The second limitation stems from the fact that the proof requires to iteratively define efficient functions  $F_i$ , where each  $F_i$  builds upon an (efficient) OWF inverter applied to  $F_{i-1}$ . The total number of functions can be picked to be the minimum of the number of queries to the OWF made by either of the two parties. However, this argument crucially relies on the fact that the number of functions  $F_i$  is a constant; to see this, imagine that we have at our disposal a OWF inverter that would always makes a number of queries quadratic in the number of queries made by the function in the forward direction. Such an inverter would be efficient (i.e., it inverses any poly time function in poly time), yet one cannot obtain a poly time function by iteratively defining a function  $F_i$  which invokes  $\text{Inv}_{F_{i-1}}$  unless  $i$  is constant, since the complexity of  $F_i$  grows as  $\text{runtime}(F_1)^{2^i}$ .

Eventually, our result in this section focuses on perfectly correct key agreement. We discuss the case of imperfect key agreement in Section 4.2.

#### 4.1.1 A Helpful Logical Lemma

Below, we state a simple lemma which allows for more direct proofs of [semi  $\rightarrow$   $\forall\exists$ -semi] black-box uselessness.

► **Lemma 19.** *Let  $\mathcal{P}$  and  $\mathcal{Q}$  be two primitives. Then whenever the following statement is established, it implies in particular that  $\mathcal{Q}$  is [semi  $\rightarrow$   $\forall\exists$ -semi] black-box useless for  $\mathcal{P}$ :*

*“Fix any primitive  $\mathcal{Z}$  and any  $Z \in F_Z$ . Assume that there exists an oracle-aided PPT  $P_1$  such that for any  $Q \in F_Q$ ,  $P_1^{Q,Z} \in F_P$ . Further assume that whenever  $(Q, Z)$  is a secure implementation of  $(\mathcal{Q}, \mathcal{Z})$ , then  $P_1^{Q,Z}$  is a secure implementation of  $\mathcal{P}$ . Then there exists an efficient implementation  $P_2^Z$  of  $\mathcal{P}$  relative to  $Z$ , and furthermore, whenever  $Z$  is a secure implementation of  $\mathcal{Z}$ ,  $P_2^Z$  is a secure implementation of  $\mathcal{P}$ .”*

We prove this lemma in the full version of the paper.

#### 4.1.2 Proof of Theorem 17

Let io- $\mathcal{F}$  be the io-OWF primitive. To prove of Theorem 17, we will prove the following:

► **Lemma 20.** *Fix any primitive  $\mathcal{Z}$  and any  $Z \in F_Z$ . Assume that there exists an oracle-aided PPT  $KA_1$  such that for any implementation ioF of an infinitely-often one-way function,  $KA_1^{\text{ioF}, Z}$  implements an infinitely-often perfect key agreement unbalanced with respect to ioF, relative to  $(\text{ioF}, Z)$ . Assume furthermore that if  $(\text{ioF}, Z)$  is a secure implementation of  $(\text{io-}\mathcal{F}, \mathcal{Z})$ , then  $KA_1^{\text{ioF}, Z}$  is a secure implementation of infinitely-often key agreement, unbalanced with respect to ioF. Then there exists an efficient implementation  $KA_2$  of (infinitely-often) key agreement relative to  $Z$ , and furthermore, if  $Z$  is a secure implementation of  $\mathcal{Z}$ , then  $KA_2^Z$  is a secure implementation of infinitely-often key agreement.*



The proof of Theorem 17 follows directly from the above lemma by applying Lemma 19. To prove Lemma 20, we rely on the following lemma.

► **Lemma 21.** *Let  $\text{RO}$  be a random oracle. For any auxiliary oracle  $\mathcal{Z}$ , if there exists no infinitely-often one-way function relative to  $\mathcal{Z}$ , then there exists no construction  $\text{KA}^{\text{RO}, \mathcal{Z}}$  of a perfect infinitely-often key agreement which is unbalanced with respect to  $\text{RO}$ .*

Given Lemma 21, the proof of Lemma 20 follows from a disjunction argument: fix any auxiliary primitive  $\mathcal{Z}$  and any  $Z \in F_{\mathcal{Z}}$ . Two complementary cases can occur:

- Either there exists an efficient implementation of an infinitely-often one-way function  $\text{ioF}^Z$  relative to  $\mathcal{Z}$ . By the assumption of Lemma 20, there exists an efficient implementation  $\text{KA}_1$  of key agreement relative to  $(\text{ioF}', \mathcal{Z})$  for any  $\text{ioF}' \in F_{\text{io-}\mathcal{F}}$ . Define the following efficient construction  $\text{KA}_2^Z$ :  $\text{KA}_2^Z := \text{KA}_1^{\text{ioF}^Z, \mathcal{Z}}$ . By our assumption, this is therefore an efficient implementation of key agreement relative to  $\mathcal{Z}$ , which is also secure if  $(\text{ioF}, \mathcal{Z})$  is secure.
- Or there exists no efficient implementation of an infinitely-often one-way function  $\text{ioF}^Z$  relative to  $\mathcal{Z}$ . By Lemma 21, for a random oracle  $\text{RO}$ , there must therefore exist an efficient attack on  $\text{KA}_1^{\text{RO}, \mathcal{Z}}$ . By Theorem 5.2 of [20], with measure 1 over the choice of the random oracle  $\text{RO}$ ,  $\text{RO}$  is a one-way function (and therefore in particular an  $\text{io-OWF}$ ; note that this theorem holds also in the presence of an arbitrary other oracle). Therefore,  $\text{KA}_1$  is not a secure implementation of key agreement with respect to any  $(\text{ioF}', \mathcal{Z})$  with  $\text{ioF}' \in F_{\text{io-}\mathcal{F}}$ , and by the assumptions of Lemma 21,  $(\text{RO}, \mathcal{Z})$  is not a secure implementation of  $(\text{io-}\mathcal{F}, \mathcal{Z})$ . Since  $\text{RO}$  is a secure implementation of  $\text{io-OWF}$ , this implies that  $\mathcal{Z}$  is not a secure implementation of  $\mathcal{Z}$ . Therefore, we can define  $\text{KA}_2$  to be the trivial protocol in which Alice samples the output key and sends it to Bob. This is an efficient implementation of key agreement (it need not be secure since  $\mathcal{Z}$  is not a secure implementation of  $\mathcal{Z}$ ).

### 4.1.3 Proof of Lemma 21

It remains to prove Lemma 21. Consider a candidate construction  $\text{KA}^{\text{RO}, \mathcal{Z}}$  of key agreement such that one of Alice and Bob makes a constant number of queries to  $\text{RO}$ . Let  $\lambda \in \mathbb{N}$  be the security parameter, and consider a run  $(T, K_A, K_B) \xleftarrow{\$} \langle \text{Alice}^{\text{RO}, \mathcal{Z}}(1^\lambda), \text{Bob}^{\text{RO}, \mathcal{Z}}(1^\lambda) \rangle$  of the construction  $\text{KA}^{\text{RO}, \mathcal{Z}}$ . We will describe an efficient attacker  $\text{Eve}^{\text{RO}, \mathcal{Z}}$  that breaks  $\text{KA}^{\text{RO}, \mathcal{Z}}$  for infinitely many  $\lambda$ . The attack closely follows the (inefficient) strategy of [9], but relies on  $\text{Inv}^Z$  to make the attack efficient. Without loss of generality, assume that Bob makes a constant number of queries; let  $q_B$  be a constant bound on the number of queries made by Bob in any execution of the protocol. Furthermore, let  $r_A(\lambda)$  and  $r_B(\lambda)$  be (polynomial) bounds on the length of the random tape of Alice and Bob respectively, and let  $q(\lambda) = q_A(\lambda) + q_B$  be a (polynomial) bound on the total number of queries to  $\text{RO}$  made by both parties in any execution.

#### 4.1.3.1 Lazy oracle sampling

Let  $q \in \mathbb{N}$ . For any string  $r$  of length  $q$ , and any list  $L$  of (query, answer) pairs, we let  $\text{SimRO}[L]_q(\cdot; r)$  be a stateful *lazy sampler* for a random oracle consistent with  $L$ . Namely,  $\text{SimRO}[L]_q(\cdot; r)$  works as follows: it maintains a counter  $i$  (initialized to 1) and a list  $L'$  of pairs (query, answer), which is initially empty. Each time it receives an input  $x$ ,  $\text{SimRO}[L]_q(x; r)$  first checks whether or not the query belongs to  $L \cup L'$ , and outputs the corresponding answers if this holds. If the query does not belong to  $L$  or  $L'$ , algorithm  $\text{SimRO}[L]_q(x; r)$  defines  $q_i$  to be the answer to the query, adds (query,  $q_i$ ) to  $L'$ , and sets  $i \leftarrow i + 1$ . Note that for any interactive protocol  $\Pi^{\text{RO}}$  where the parties make less than  $q$  queries in total,

and any list  $L$  of (query, answer) pairs consistent with  $\text{RO}$ , the distribution of the views of all parties obtained by sampling a random oracle  $\text{RO}$  and running  $\Pi^{\text{RO}}$  is identical to the distribution of the views of all parties obtained by sampling a  $q$ -bit string  $r$  and running  $\Pi$  while emulating  $\text{RO}$  using  $\text{SimRO}[L]_q(\cdot; r)$ .

#### 4.1.3.2 An inefficient attack

We first describe an inefficient attack on the candidate construction  $\text{KA}^{\text{RO}, \text{Z}}$ , taken almost verbatim from [9]. The attacker  $\text{Eve}^{\text{RO}, \text{Z}}$ , given a transcript  $T$  of an execution of  $\text{KA}^{\text{RO}, \text{Z}}$ , maintains a set  $Q_E$  of query/answer pairs for  $\text{Z}$ , and a multi-set of candidate keys  $\mathcal{K}$ , both initialized to  $\emptyset$ . Eve runs  $2q_B(\lambda) + 1$  iterations of the following procedure.

- *Simulation phase:* Eve finds a view of  $\text{Alice}^{\text{RO}', \text{Z}}$  with respect to some (possibly different) oracle  $\text{RO}'$ , consistent with the transcript  $T$  and all pairs query/answer in  $Q_E$ . This view contains a random tape  $r_A$ , the set of queries  $Q_A$  made by  $\text{Alice}^{\text{RO}', \text{Z}}$  (which is consistent with  $Q_{\text{Eve}}$ , but not necessarily with  $\text{RO}$ ), and the key  $K_A$  computed by Alice. Eve adds  $K_A$  to  $\mathcal{K}$ .
- *Update phase:*  $\text{Eve}^{\text{RO}, \text{Z}}$  makes all queries in  $Q_A$  to the true random oracle  $\text{RO}$ , and adds the results to  $Q_E$ .

After running  $2q_B(\lambda) + 1$  iterations of the above attack, Eve has a multi-set  $\mathcal{K}$  of  $2q_B + 1$  possible keys; Eve outputs the majority value in  $\mathcal{K}$ . Observe that during each round of the attack, two events can happen:

1. Either one of the new queries (not already contained in  $Q_E$ ) made by Alice in the simulated run was made by Bob in the real execution of the protocol. In this case, Eve discovers (and adds to  $Q_E$ ) a new query of Bob.
2. Or none of the new queries of Alice was made by Bob in the real protocol, in which case there exists an oracle  $\text{RO}'$  which is consistent with the view of Bob in the real protocol, and the view of Alice in the simulated run. By perfect correctness, this means that the key  $K_A$  computed by Alice in this run is necessarily the correct key.

Now, since Bob makes at most  $q_B$  distinct queries, the first of the two events can happen at most  $q_B$  times, hence the second event necessarily happens at least  $q_B + 1$  times, which guarantees that the majority value in the multi-set  $\mathcal{K}$  is indeed the correct key with probability 1.

The above attack requires  $\mathcal{O}(q_A q_B)$  queries to  $\text{RO}$ . However, it requires to find a view for  $\text{Alice}^{\text{RO}', \text{Z}}$  consistent with a given transcript, where  $\text{RO}'$  is a simulated random oracle, but  $\text{Z}$  is a “true” auxiliary oracle. In general, this might require exponentially many queries to  $\text{Z}$ , hence  $\text{Eve}^{\text{RO}, \text{Z}}$  is not an efficient oracle-aided algorithm. In the following, we show how to make the attack efficient given an inverter for io-OWFs.

#### 4.1.3.3 Lazy protocol emulation

Let  $L$  be a list of (query, answer) pairs to  $\text{RO}$ . Given the construction  $\text{KA}^{\text{RO}, \text{Z}}$ , let  $\text{SimC}_L^{\text{Z}}$  be an oracle-aided PPT algorithm that emulates a run of Alice and Bob in  $\text{KA}^{\text{RO}, \text{Z}}$  that is consistent with  $L$  (but not necessarily with the rest of  $\text{RO}$ ). That is, given a random string  $r_A || r_B || q$  of length  $r_A(\lambda) + r_B(\lambda) + q(\lambda)$ ,  $\text{SimC}_L^{\text{Z}}(1^\lambda; r_A || r_B || q)$  does the following: it runs Alice and Bob with input  $1^\lambda$  and respective random tapes  $r_A$  and  $r_B$ , while using  $\text{SimRO}[L]_q(\cdot; q)$  to lazily emulate the random oracle  $\text{RO}$ . After completion of the protocol,  $\text{SimC}$  outputs the transcript  $T$  of the interaction, the lists  $(Q_A, Q_B)$  of all queries to  $\text{RO}$  made by Alice and Bob during the emulation of the protocol (together with their answers), and the outputs  $(K_A, K_B)$  of both parties. Observe that  $\text{SimC}$  corresponds to a valid interaction between  $\text{Alice}^{\text{RO}'}(1^\lambda; r_A)$  and  $\text{Bob}^{\text{RO}'}(1^\lambda; r_B)$  with respect to a random oracle  $\text{RO}'$  sampled uniformly at random, conditioned on being consistent with  $L$ .

#### 4.1.3.4 The inverter

Since there is no infinitely-often OWF relative to  $Z$ , there exists an efficient inverter for any efficient oracle-aided function:

► **Lemma 22.** *For any oracle-aided PPT function  $F^Z$  and any polynomial  $p$ , there exists an oracle-aided PPT inverter  $\text{Inv}_{F,p}^Z$  such that for all large enough  $n \in \mathbb{N}$ , it holds that*

$$\Pr_{x \leftarrow \{0,1\}^n} [\text{Inv}_{F,p}^Z(F^Z(x), 1^n) \in (F^{-1})^Z(F^Z(x))] \geq 1 - \frac{1}{p(n)}.$$

**Proof.** This follows directly from the fact that the inexistence of io-OWFs relative to  $Z$  implies the inexistence of *weak* io-OWFs relative to  $Z$  (a weak OWF is a OWF where security is relaxed by saying that there exists a polynomial  $p$  such that no efficient adversary can invert with probability better than  $1 - 1/p(n)$ ). The latter follows from standard hardness amplification methods as was initially proven by Yao [29]. ◀

#### 4.1.3.5 The sequence of functions

Let  $p : \lambda \rightarrow 1/(6q_B + 3)$  be a constant polynomial. We iteratively define a sequence of  $2(q_B + 1)$  oracle functions  $(F_0^Z, G_0^Z), \dots, (F_{q_B}^Z, G_{q_B}^Z)$  as follows.

- $F_0^Z$  gets as input a string  $(r_A || r_B || q_0)$  of length  $r_A(\lambda) + r_B(\lambda) + q(\lambda)$ , computes
 
$$(T, Q_A, Q_B, K_A, K_B) \leftarrow \text{SimC}_\emptyset^Z(1^\lambda; r_A || r_B || q),$$
 and outputs  $T$ . The function  $G_0^Z$  is defined similarly, but outputs  $(T, Q_A, Q_B, K_A)$ .
- $F_1^Z$  gets as input a string  $(r_A || r_B || q_0 || q_1)$  of length  $r_A(\lambda) + r_B(\lambda) + 2q(\lambda)$ . First, it computes  $(T, Q_A, Q_B, K_A) \leftarrow G_0^Z(r_A || r_B || q_0)$ . Second, it sets  $n \leftarrow r_A(\lambda) + r_B(\lambda) + q(\lambda)$  and runs  $(r'_A || r'_B || q'_0) \leftarrow \text{Inv}_{F_0,p}^Z(T, 1^n)$ . Third, it computes  $(T', Q'_A, Q'_B, K'_A) \leftarrow G_0^Z(1^\lambda; r'_A || r'_B || q'_0)$ . Eventually, it uses  $\text{SimRO}[Q_A \cup Q_B](\cdot; q_1)$  to lazily sample the answers to all queries contained in  $Q'_A$ , and stores the results in a set  $Q_E$  of pairs query/answer.  $F_1^Z$  outputs  $(T, Q_E)$ . We also define  $G_1^Z$  to be the function defined as  $F_1^Z$  except that it additionally outputs  $(Q_A, Q_B, K'_A)$ .
- $F_i^Z$  gets as input a string  $(r_A || r_B || q_0 || \dots || q_i)$  of length  $r_A(\lambda) + r_B(\lambda) + (i + 1) \cdot q(\lambda)$ . First, it computes  $(T, Q_E, Q_A, Q_B) \leftarrow G_{i-1}^Z(r_A || r_B || q_0 || \dots || q_{i-1})$ . Second, it sets  $n \leftarrow r_A(\lambda) + r_B(\lambda) + i \cdot q(\lambda)$  and runs  $(r'_A || r'_B || q'_0 || \dots || q'_{i-1}) \leftarrow \text{Inv}_{F_{i-1},p}^Z((T, L_{i-1}), 1^n)$ . Third, it computes  $(T', Q'_A, Q'_B) \leftarrow G_0^Z(1^\lambda; r'_A || r'_B || q'_0)$ . Eventually, it uses  $\text{SimRO}[Q_A \cup Q_B](\cdot; q_i)$  to lazily sample the answers to all queries contained in  $Q'_A$ , and add the results to  $Q_E$ .  $F_i^Z$  outputs  $(T, Q_E)$ . We also define  $G_i^Z$  to be the function defined as  $F_i^Z$  except that it additionally outputs  $(Q_A, Q_B, K'_A)$ .

For readability, we also provide a pseudocode for the function  $F_i^Z$  below:

```

function  $F_i^Z(r_A || r_B || q_0 || \dots || q_i)$   $\triangleright (r_A || r_B || q_1 || \dots || q_i)$  is of length
 $r_A(\lambda) + r_B(\lambda) + (i + 1) \cdot q(\lambda)$ 
   $(T, Q_E, Q_A, Q_B, K_A) \leftarrow G_{i-1}^Z(r_A || r_B || q_1 || \dots || q_{i-1})$ 
   $n \leftarrow r_A(\lambda) + r_B(\lambda) + i \cdot q(\lambda)$ 
   $(r'_A || r'_B || q'_0 || \dots || q'_{i-1}) \leftarrow \text{Inv}_{F_{i-1},p}^Z((T, Q_E), 1^n)$   $\triangleright p : \lambda \rightarrow 1/(6q_B + 3)$ 
   $(T', Q'_A, Q'_B, K'_A) \leftarrow G_0^Z(1^\lambda; r'_A || r'_B || q'_0)$ 
  for  $(x, y) \in Q'_A$  do
     $Q_E \leftarrow Q_E \cup (x, \text{SimRO}[Q_A \cup Q_B](x; q_i))$ 
  end for
  return  $(T, Q_E)$   $\triangleright G_i^Z$  is similar but additionally outputs  $(Q_A, Q_B, K'_A)$ 
end function

```

#### 4.1.3.6 Making the [9] attack efficient with Inv

To overcome the inefficiency of the attack of [9], we leverage the fact that, by assumption, there exists no infinitely-often one-way function relative to  $Z$ . As before, the attacker  $\text{newEve}^{\text{RO}, Z}$ , given a transcript  $T$  of an execution of  $\text{KA}^{\text{RO}, Z}$ , maintains a set  $Q_E$  of query/answer pairs for  $Z$ , and a multi-set of candidate keys  $\mathcal{K}$ , both initialized to  $\emptyset$ . Let  $p : \lambda \rightarrow 1/(6q_B + 3)$  be a constant polynomial.  $\text{newEve}$  runs  $2q_B + 1$  iterations of the following attack.

- *Simulation phase:* During the  $i$ -th round of attack,  $\text{newEve}$  does the following:
  1. *Finding a view of Alice consistent with  $T$  and  $Q_E$ :*  $\text{newEve}$  sets  $n \leftarrow r_A(\lambda) + r_B(\lambda) + i \cdot q(\lambda)$  and computes  $(r'_A || r'_B || q'_0 || \dots || q'_{i-1}) \leftarrow \text{Inv}_{F_{i-1}, p}^Z((T, Q_E); 1^n)$ .
  2. *Simulating the run of Alice with the view above:*  $\text{newEve}$  computes  $(T', Q'_A, Q'_B, K'_A) \leftarrow G_1^Z(1^\lambda; r'_A || r'_B || q'_0)$ .
  3. *Storing the key:*  $\text{newEve}$  adds  $K'_A$  to  $\mathcal{K}$ .
- *Update phase:*  $\text{Eve}^{\text{RO}, Z}$  makes all queries in  $Q'_A$  to the true random oracle  $\text{RO}$ , and adds the results to  $Q_E$ .

After running  $2q_B + 1$  iterations of the above attack,  $\text{Eve}$  has a multi-set  $\mathcal{K}$  of  $2q_B + 1$  possible keys;  $\text{Eve}$  outputs the majority value in  $\mathcal{K}$ . We now analyze the success probability of the attack.

▷ **Claim 23.**  $\text{newEve}$  outputs the correct key with probability at least  $2/3$ .

First, observe that by definition of  $F_0^Z$ , the distribution of transcripts  $T$  of the real execution of the protocol (which  $\text{newEve}$  gets as input) is distributed exactly as  $F_0^Z(r_A || r_B || q_0)$  for uniformly random  $(r_A, r_B, q_0)$ , where  $r_A$  (resp.,  $r_B$ ) is the real random tape of  $\text{Alice}^{\text{RO}, Z}$  (resp.,  $\text{Bob}^{\text{RO}, Z}$ ) and  $q_0$  is the ordered string of all answers of  $\text{RO}$  to distinct queries from Alice and Bob. Therefore, by definition of  $\text{Inv}_{F_0, p}^Z$ , the tuple  $(r'_A || r'_B || q'_0)$  computed in the first iteration of the attack is consistent with the real transcript  $T$  with probability at least  $p = 1/(6q_B + 3)$ .

Consider now the set  $Q_E$  of queries obtained by  $\text{newEve}$  after the update phase of the first iteration.  $(T, Q_E)$  is distributed exactly as  $F_1^Z(r_A || r_B || q_0 || q_1)$  for uniformly random  $(r_A, r_B, q_0, q_1)$ . This is because  $Q_E$  in  $F_1^Z$  is computed by lazily sampling the answers of a random oracle, conditioned on being consistent with all queries made by Alice and Bob in the execution of  $F_0^Z(r_A || r_B || q_0)$ . Since the real run of the protocol corresponds to an execution of  $F_0^Z$  on a random input  $(r_A || r_B || q_0)$ , and making the queries in  $Q'_A$  to  $\text{RO}$  is identical to lazily sampling  $\text{RO}$  while being consistent with  $q_0$  (i.e., the query/answer pairs obtained by Alice and Bob in the real execution). Therefore, the tuple  $(r'_A || r'_B || q'_0 || q'_1)$  which  $\text{newEve}$  computes in the second iteration of the attack is consistent with the real transcript  $T$  with probability at least  $p = 1/(6q_B + 3)$ .

More generally, the distribution of  $(T, Q_E)$  obtained by  $\text{newEve}$  during the  $i$ -th iteration of the attack after receiving the transcript  $T$  of a real execution of the protocol is distributed exactly as  $F_i^Z(r_A || r_B || q_0 || \dots || q_i)$  for uniformly random  $(r_A, r_B, q_0, \dots, q_i)$ , hence the tuple  $(r'_A || r'_B || q'_0 || \dots || q'_i)$  which  $\text{newEve}$  computes in the  $(i+1)$ -th iteration of the attack is consistent with the real transcript  $T$  with probability at least  $p = 1/(6q_B + 3)$ .

Putting everything together, after finishing the attack, by a straightforward union bound, all simulated views computed by  $\text{newEve}$  during the attack are consistent with  $T$  with probability at least  $1 - (2q_B + 1) \cdot 1/(6q_B + 3) = 2/3$ . When this happens, by the same argument as for the inefficient attack, the majority key in  $\mathcal{K}$  is necessarily the correct key, and the claim follows.

▷ **Claim 24.** The number of queries made by  $\text{newEve}$  to  $\text{RO}$  and  $Z$  is bounded by a polynomial.

As in the inefficient attack, `newEve` makes at most  $\mathcal{O}(q_A q_B) = \mathcal{O}(q_A)$  queries to RO. The polynomial bound on the number of queries to  $Z$  follows from the efficiency of  $\text{Inv}$ :  $\text{Inv}_{F_0,p}^Z$  is efficient by definition, hence  $F_1^Z$  (which invokes  $\text{Inv}_{F_0,p}^Z$  internally) is an efficient function, from which we get that  $\text{Inv}_{F_1,p}^Z$  is also efficient, and so on, and the claim follows.

## 4.2 Black-Box Uselessness of OWFs for Imperfect KA

In this section, we discuss how our result of the previous section can be extended to the case of imperfect key agreement. More precisely, we provide a sketch of how to modify our previous proof to show the following:

- **Theorem 25.** *Infinitely-often one-way functions are [semi  $\rightarrow \forall\exists$ -semi] black-box useless for infinitely-often perfect key agreement in any construction where:*
- *Both parties make a constant number of queries to the io-OWF (but any polynomial number of queries to the auxiliary oracle)*
  - *The key agreement protocol has a constant number of rounds.*

**Proof Sketch.** The natural approach to extend our result is to replace the attack of [9] by an attack that applies to any imperfect key agreement protocol in the random-oracle model, such as those of Impagliazzo and Rudich [20] or Barak and Mahmoody [4], making the same case distinction based on the existence of io-OWFs to make the attack efficient. The structure of these attacks are very similar, though more involved than the attack of [9]: they proceed in a sequence of steps, where each step has a simulation phase, in which the attacker samples views consistent with (a portion of) the transcript and a set of queries, and an update phase, where the attacker makes some queries based on the simulated run.

Two important technicalities arise when modifying our previous proof with the attacks of [4, 20]:

First, in the simpler attack of [9], the perfect correctness guarantees that finding *any* consistent view is sufficient; in the attacks of [4, 20], however, the attacker is required to *sample* views from a distribution close to the uniform distribution over views conditioned on a transcript and a set of queries. This can still be achieved assuming only the inexistence of io-OWFs: the inexistence of io-OWFs further entails the inexistence of *distributional* io-OWFs [19]. Namely, we must rely on the following lemma, a proof of which can be found in [6].

- **Lemma 26.** *Assume that there exists no infinitely-often one-way function relative to  $Z$ . Then for any efficient oracle-aided function  $F^Z$  and any polynomial  $p$ , there exists an inverter  $\text{Inv}_F^Z$  such that for all large enough  $n \in \mathbb{N}$ ,*

$$\Pr_{x \xleftarrow{\$} \{0,1\}^n, y \leftarrow F^Z(x)} \left[ \text{SD} \left( (F^{-1})^Z(y), \text{Inv}_F^Z(y, 1^n) \right) > \frac{1}{p(n)} \right] \leq \frac{1}{p(n)},$$

where  $\text{SD}$  denotes the statistical distance between the two distributions.

Second, the attacks of [4, 20] proceed in a number of steps that grows with the number of queries of *both* parties (to the random oracle) *and* the round complexity of the protocol. More precisely, the attack requires executing several simulation and updates phases (of the order of  $\tilde{\mathcal{O}}(q_A q_B)$ , where  $q_A, q_B$  bound the respective number of queries of Alice and Bob to the random oracle) for each round of the protocol, where the simulation phase for the round  $i$  inverse samples views consistent with the set of queries made by the attacker so far and the transcript of the protocol *up to round  $i$* . This means that to be carried efficiently, the key agreement must be constant round, and both parties must make a constant number of queries to the random oracle.

From here, a proof of Theorem 25 follows by fixing a (constant) bound  $B$  on the total number of steps of the (inefficient) attacker, and using the inverse sampler guaranteed by Lemma 26 for statistical distance  $1/(10B)$  to make it efficient, similarly as in our previous proof. By a union bound over all steps, the  $B$  steps of the inverse sampling will simultaneously guarantee that, with probability at least  $1/10$ , at any round  $i$ , no intersection query between Alice and Bob made prior to round  $i$  has been missed by the attacker. This allows us to conclude that the overall success probability of the efficient attacker (which uses the inverse sampler) is at most a tenth of the success probability of the inefficient attacker described in [4, 20].

With these technicalities in mind, this proof shows that io-OWFs are  $[\text{semi} \rightarrow \forall\exists\text{-semi}]$  black-box useless for constant-query constant-round constructions of imperfect key agreement. ◀

---

## References

- 1 Shashank Agrawal, Venkata Koppula, and Brent Waters. Impossibility of simulation secure functional encryption even with random oracles. In *Theory of Cryptography Conference*, pages 659–688. Springer, 2018.
- 2 Gilad Asharov and Gil Segev. Limits on the power of indistinguishability obfuscation and functional encryption. In Venkatesan Guruswami, editor, *56th Annual Symposium on Foundations of Computer Science*, pages 191–209, Berkeley, CA, USA, October 17–20 2015. IEEE Computer Society Press. doi:10.1109/FOCS.2015.21.
- 3 Paul Baecher, Christina Brzuska, and Marc Fischlin. Notions of black-box reductions, revisited. In Kazue Sako and Palash Sarkar, editors, *Advances in Cryptology – ASIACRYPT 2013, Part I*, volume 8269 of *Lecture Notes in Computer Science*, pages 296–315, Bangalore, India, December 1–5 2013. Springer, Heidelberg, Germany. doi:10.1007/978-3-642-42033-7\_16.
- 4 Boaz Barak and Mohammad Mahmoody-Ghidary. Merkle puzzles are optimal - an  $O(n^2)$ -query attack on any key exchange from a random oracle. In Shai Halevi, editor, *Advances in Cryptology – CRYPTO 2009*, volume 5677 of *Lecture Notes in Computer Science*, pages 374–390, Santa Barbara, CA, USA, August 16–20 2009. Springer, Heidelberg, Germany. doi:10.1007/978-3-642-03356-8\_22.
- 5 Balthazar Bauer, Pooya Farshim, and Sogol Mazaheri. Combiners for backdoored random oracles. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018, Part II*, volume 10992 of *Lecture Notes in Computer Science*, pages 272–302, Santa Barbara, CA, USA, August 19–23 2018. Springer, Heidelberg, Germany. doi:10.1007/978-3-319-96881-0\_10.
- 6 Itay Berman, Iftach Haitner, and Aris Tentes. Coin flipping of *any* constant bias implies one-way functions. In David B. Shmoys, editor, *46th Annual ACM Symposium on Theory of Computing*, pages 398–407, New York, NY, USA, May 31 – June 3 2014. ACM Press. doi:10.1145/2591796.2591845.
- 7 Nir Bitansky, Huijia Lin, and Omer Paneth. On removing graded encodings from functional encryption. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 3–29. Springer, 2017.
- 8 Zvika Brakerski, Christina Brzuska, and Nils Fleischhacker. On statistically secure obfuscation with approximate correctness. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016, Part II*, volume 9815 of *Lecture Notes in Computer Science*, pages 551–578, Santa Barbara, CA, USA, August 14–18 2016. Springer, Heidelberg, Germany. doi:10.1007/978-3-662-53008-5\_19.
- 9 Zvika Brakerski, Jonathan Katz, Gil Segev, and Arkady Yerukhimovich. Limits on the power of zero-knowledge proofs in cryptographic constructions. In Yuval Ishai, editor, *TCC 2011: 8th Theory of Cryptography Conference*, volume 6597 of *Lecture Notes in Computer Science*, pages 559–578, Providence, RI, USA, March 28–30 2011. Springer, Heidelberg, Germany. doi:10.1007/978-3-642-19571-6\_34.



- 10 Ran Canetti, Yael Tauman Kalai, and Omer Paneth. On obfuscation with random oracles. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015: 12th Theory of Cryptography Conference, Part II*, volume 9015 of *Lecture Notes in Computer Science*, pages 456–467, Warsaw, Poland, March 23–25 2015. Springer, Heidelberg, Germany. doi:10.1007/978-3-662-46497-7\_18.
- 11 Kai-Min Chung, Huijia Lin, Mohammad Mahmoody, and Rafael Pass. On the power of nonuniformity in proofs of security. In Robert D. Kleinberg, editor, *ITCS 2013: 4th Innovations in Theoretical Computer Science*, pages 389–400, Berkeley, CA, USA, January 9–12 2013. Association for Computing Machinery. doi:10.1145/2422436.2422480.
- 12 Sanjam Garg, Mohammad Hajiabadi, Mohammad Mahmoody, and Ameer Mohammed. Limits on the power of garbling techniques for public-key encryption. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018, Part III*, volume 10993 of *Lecture Notes in Computer Science*, pages 335–364, Santa Barbara, CA, USA, August 19–23 2018. Springer, Heidelberg, Germany. doi:10.1007/978-3-319-96878-0\_12.
- 13 Sanjam Garg, Mohammad Mahmoody, and Ameer Mohammed. Lower bounds on obfuscation from all-or-nothing encryption primitives. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017, Part I*, volume 10401 of *Lecture Notes in Computer Science*, pages 661–695, Santa Barbara, CA, USA, August 20–24 2017. Springer, Heidelberg, Germany. doi:10.1007/978-3-319-63688-7\_22.
- 14 Sanjam Garg, Mohammad Mahmoody, and Ameer Mohammed. Lower bounds on obfuscation from all-or-nothing encryption primitives. In *Draft of the full version*, 2017. URL: <http://www.cs.virginia.edu/~mohammad/files/papers/I0-all-or-nothing.pdf>.
- 15 Sanjam Garg, Mohammad Mahmoody, and Ameer Mohammed. When does functional encryption imply obfuscation? In Yael Kalai and Leonid Reyzin, editors, *TCC 2017: 15th Theory of Cryptography Conference, Part I*, volume 10677 of *Lecture Notes in Computer Science*, pages 82–115, Baltimore, MD, USA, November 12–15 2017. Springer, Heidelberg, Germany. doi:10.1007/978-3-319-70500-2\_4.
- 16 Rosario Gennaro, Yael Gertner, Jonathan Katz, and Luca Trevisan. Bounds on the efficiency of generic cryptographic constructions. *SIAM journal on Computing*, 35(1):217–246, 2005.
- 17 Iftach Haitner, Jonathan J. Hoch, Omer Reingold, and Gil Segev. Finding collisions in interactive protocols - a tight lower bound on the round complexity of statistically-hiding commitments. In *48th Annual Symposium on Foundations of Computer Science*, pages 669–679, Providence, RI, USA, October 20–23 2007. IEEE Computer Society Press. doi:10.1109/FOCS.2007.27.
- 18 Justin Holmgren and Alex Lombardi. Cryptographic hashing from strong one-way functions (or: One-way product functions and their applications). In Mikkel Thorup, editor, *59th Annual Symposium on Foundations of Computer Science*, pages 850–858, Paris, France, October 7–9 2018. IEEE Computer Society Press. doi:10.1109/FOCS.2018.00085.
- 19 Russell Impagliazzo and Michael Luby. One-way functions are essential for complexity based cryptography (extended abstract). In *30th Annual Symposium on Foundations of Computer Science*, pages 230–235, Research Triangle Park, NC, USA, October 30 – November 1 1989. IEEE Computer Society Press. doi:10.1109/SFCS.1989.63483.
- 20 Russell Impagliazzo and Steven Rudich. Limits on the provable consequences of one-way permutations. In *21st Annual ACM Symposium on Theory of Computing*, pages 44–61, Seattle, WA, USA, May 15–17 1989. ACM Press. doi:10.1145/73007.73012.
- 21 Mohammad Mahmoody and Ameer Mohammed. On the power of hierarchical identity-based encryption. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016, Part II*, volume 9666 of *Lecture Notes in Computer Science*, pages 243–272, Vienna, Austria, May 8–12 2016. Springer, Heidelberg, Germany. doi:10.1007/978-3-662-49896-5\_9.



- 22    Mohammad Mahmoody, Ameer Mohammed, Soheil Nematihaji, Rafael Pass, and abhi shelat. Lower bounds on assumptions behind indistinguishability obfuscation. In Eyal Kushilevitz and Tal Malkin, editors, *TCC 2016-A: 13th Theory of Cryptography Conference, Part I*, volume 9562 of *Lecture Notes in Computer Science*, pages 49–66, Tel Aviv, Israel, January 10–13 2016. Springer, Heidelberg, Germany. doi:10.1007/978-3-662-49096-9\_3.
- 23    Mohammad Mahmoody, Ameer Mohammed, Soheil Nematihaji, Rafael Pass, and abhi shelat. A note on black-box separations for indistinguishability obfuscation. Cryptology ePrint Archive, Report 2016/316, 2016. URL: <http://eprint.iacr.org/2016/316>.
- 24    Mohammad Mahmoody, Tal Moran, and Salil P. Vadhan. Time-lock puzzles in the random oracle model. In Phillip Rogaway, editor, *Advances in Cryptology – CRYPTO 2011*, volume 6841 of *Lecture Notes in Computer Science*, pages 39–50, Santa Barbara, CA, USA, August 14–18 2011. Springer, Heidelberg, Germany. doi:10.1007/978-3-642-22792-9\_3.
- 25    Hemanta K. Maji and Mingyuan Wang. Black-box use of one-way functions is useless for optimal fair coin-tossing. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology – CRYPTO 2020, Part II*, volume 12171 of *Lecture Notes in Computer Science*, pages 593–617, Santa Barbara, CA, USA, August 17–21 2020. Springer, Heidelberg, Germany. doi:10.1007/978-3-030-56880-1\_21.
- 26    Omer Reingold, Luca Trevisan, and Salil P. Vadhan. Notions of reducibility between cryptographic primitives. In Moni Naor, editor, *TCC 2004: 1st Theory of Cryptography Conference*, volume 2951 of *Lecture Notes in Computer Science*, pages 1–20, Cambridge, MA, USA, February 19–21 2004. Springer, Heidelberg, Germany. doi:10.1007/978-3-540-24638-1\_1.
- 27    Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In David B. Shmoys, editor, *46th Annual ACM Symposium on Theory of Computing*, pages 475–484, New York, NY, USA, May 31 – June 3 2014. ACM Press. doi:10.1145/2591796.2591825.
- 28    Daniel R. Simon. Finding collisions on a one-way street: Can secure hash functions be based on general assumptions? In Kaisa Nyberg, editor, *Advances in Cryptology – EUROCRYPT’98*, volume 1403 of *Lecture Notes in Computer Science*, pages 334–345, Espoo, Finland, May 31 – June 4 1998. Springer, Heidelberg, Germany. doi:10.1007/BFb0054137.
- 29    Andrew Chi-Chih Yao. Theory and applications of trapdoor functions (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science*, pages 80–91, Chicago, Illinois, November 3–5 1982. IEEE Computer Society Press. doi:10.1109/SFCS.1982.45.

# Pseudobinomiality of the Sticky Random Walk

Venkatesan Guruswami

Computer Science Department, Carnegie Mellon University, Pittsburgh, PA, USA  
venkatg@cs.cmu.edu

Vinayak M. Kumar

Department of Computing and Mathematical Sciences,  
California Institute of Technology, Pasadena, CA, USA  
vmkumar@caltech.edu

---

## Abstract

Random walks on expanders are a central and versatile tool in pseudorandomness. If an arbitrary half of the vertices of an expander graph are marked, known Chernoff bounds for expander walks imply that the number  $M$  of marked vertices visited in a long  $n$ -step random walk strongly concentrates around the expected  $n/2$  value. Surprisingly, it was recently shown that the parity of  $M$  also has exponentially small bias.

Is there a common unification of these results? What other statistics about  $M$  resemble the binomial distribution (the Hamming weight of a random  $n$ -bit string)? To gain insight into such questions, we analyze a simpler model called the *sticky random walk*. This model is a natural stepping stone towards understanding expander random walks, and we also show that it is a necessary step. The sticky random walk starts with a random bit and then each subsequent bit independently equals the previous bit with probability  $(1 + \lambda)/2$ . Here  $\lambda$  is the proxy for the expander's (second largest) eigenvalue.

Using Krawtchouk expansion of functions, we derive several probabilistic results about the sticky random walk. We show an asymptotically tight  $\Theta(\lambda)$  bound on the total variation distance between the (Hamming weight of the) sticky walk and the binomial distribution. We prove that the correlation between the majority and parity bit of the sticky walk is bounded by  $O(n^{-1/4})$ . This lends hope to unifying Chernoff bounds and parity concentration, as well as establishing other interesting statistical properties, of expander random walks.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Random walks and Markov chains; Theory of computation  $\rightarrow$  Expander graphs and randomness extractors

**Keywords and phrases** Expander Graphs, Fourier analysis, Markov Chains, Pseudorandomness, Random Walks

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.48

**Funding** *Venkatesan Guruswami*: Supported by NSF grants CCF-1814603 and CCF-1908125 and a Simons Investigator Award.

*Vinayak M. Kumar*: Supported by Caltech's Summer Undergraduate Research Fellowships (SURF) Program and the Stephen Adelman Memorial Fellowship.

**Acknowledgements** V.G thanks Salil Vadhan, Vijay Bhattiprolu, Nicolas Resch, and Mahdi Cheraghchi for valuable discussions about sticky random walks at various points. Thanks to Salil also for suggesting the event underlying Theorem 18. V.K thanks Mayank Pandey for helpful discussions and pointing him to [6]. The authors thank Gil Cohen, Noam Peri, and Amnon TaShma for sharing an early version of their manuscript [2] after this work was posted at [5].

## 1 Introduction

Expander graphs and random walks on their vertices are an essential and widely employed tool in pseudorandomness, and related areas like coding theory. In this paper, we are interested in a particular simple random walk model where we mark half of the vertices of an



© Venkatesan Guruswami and Vinayak M. Kumar;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 48; pp. 48:1–48:19



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

expander graph, and look at how many marked vertices an  $n$ -step random walk visits. This model has already been explored in many influential works. In particular, expander Chernoff bounds show that the number of marked vertices visited is concentrated around  $n/2$  with exponential tails [4, 7]. In his recent breakthrough construction of  $\varepsilon$ -balanced codes, Ta-Shma [12] proved that the parity of the number of visited marked nodes has exponentially small bias. This fact is quite striking since there are distributions on  $n$  bits that are  $(n-1)$ -wise independent yet have fixed parity, so one might not expect a sensitive function like parity to exhibit such strong concentration for expander random walks in general.

Let  $\mathbf{w}$  denote the bit string indicating which steps visit a marked node in an  $n$ -step expander random walk starting at a random node. The above two results show nontrivial properties that  $\mathbf{w}$  shares with a purely random bit string. This naturally raises questions about what other statistical similarities might hold between  $\mathbf{w}$  and purely random strings. For instance, are the parity bit and majority bit of  $\mathbf{w}$  (almost) uncorrelated? This can be viewed as a unification of the above-mentioned concentration results for the high order bit and low order bit of the Hamming weight of  $\mathbf{w}$ . More generally, given some arbitrary symmetric property of  $\mathbf{w}$  (i.e., one that only depends on its Hamming weight), how does its probability deviate in an expander random walk compared to a purely random string? What is the total variation distance (TVD) between the Hamming weight distribution of  $\mathbf{w}$  and the binomial distribution? Surprisingly, to the best of our knowledge, these and many other similar questions relating to the “pseudobinomiality” properties of the weight of the sequence  $\mathbf{w}$ , seem unexplored.

Towards gaining insight into and making progress on such questions, we analyze a simpler distribution over the  $n$ -bit strings called the *sticky random walk* as a necessary stepping stone towards understanding the general expander walk. The sticky random walk  $S(n, \lambda)$  is an  $n$ -step walk on a Markov chain with two states 0, 1. The start state is chosen uniformly at random, and at each subsequent step, we stick to the same state with probability  $\frac{1+\lambda}{2}$ , and change states with probability  $\frac{1-\lambda}{2}$ . Let  $\mathbf{s}$  be the  $n$ -bit string listing the states visited by  $S(n, \lambda)$ . The sticky random walk has served as a useful proxy for analyzing the actual expander random walk. Chernoff bounds for a slight variant of the sticky random walk imply similar bounds on the expander random walk, as explicitly pointed out by Kahale [7] and also revisited in the recent work by Rao and Regev [10] who obtained tighter bounds. Such a translation, however, only works for estimating the probability of monotone events. For non-monotone functions like parity, it is not known if one can deduce bounds for the expander random walk from their counterparts for the sticky random walk.

Nevertheless, understanding the simpler sticky random walk model is a meaningful first step towards understanding the general expander walk model. In fact, it is a *necessary* step, as there are expanders where the behavior of the random walk coincides with the sticky walk (i.e., the sequences  $\mathbf{w}$  and  $\mathbf{s}$  above are identically distributed). We make this (probably folklore) relationship between these two models explicit in Section 7 (see Theorem 4 below).

## 1.1 Our Results

We answer some of the questions proposed above in the case of the sticky random walk. Our main technique is to represent (a function related to) the probability density function of the Hamming weight of the sticky random walk in the basis of Krawtchouk functions. The Krawtchouk basis is a handy choice to compare this distribution with the binomial distribution. We show the following result on the TVD between these distributions in Section 4. In all our results we think of  $\lambda$  as being fixed and the length of the walk  $n$  to be growing.

► **Theorem 1.** *The total variation distance between the weight distribution of the sticky random walk and the binomial distribution is  $\Theta(\lambda)$ .*

Note that the above shows both an upper and lower bound. For the upper bound, we show a stronger claim that an appropriately weighted  $\ell_2$  distance between the two distributions is bounded from above by  $O(\lambda^2)$  for  $\lambda$  small enough (the above then follows by Cauchy-Schwarz). However, as  $\lambda \rightarrow 1$ , the  $\ell_2$  version of the bound fails to hold, and in fact we prove that this distance grows exponentially in  $n$  for  $\lambda > 0.95$  (Section 4.2). For the lower bound, the event of the Hamming weight being in the range  $n/2 \pm \sqrt{n}$  exhibits a  $\Omega(\lambda)$  deviation in probability between these two distributions.

Our next result (in Section 5) shows that the parity and majority bits of the sticky random walk are uncorrelated (up to lower order terms).

► **Theorem 2.** *The probability that the Hamming weight of an  $n$ -step sticky random walk is even and larger than  $n/2$  is  $\frac{1}{4} + o(1)$ . The same result holds for the other three symmetric cases.*

The analysis hinges on an upper estimate for the probability that the sticky random walk has weight exactly  $\lfloor n/2 \rfloor$ , which is again obtained via the Krawtchouk expansion. We note that the arguments for the parity bias from [12] and the Chernoff bound from [4] are quite different. The above theorem hints that Krawtchouk functions might offer a more general tool that can unify these two arguments.

Next, we verify that the residues of the Hamming weight of the sticky walk with respect to any fixed modulus  $m$  are almost equidistributed (Section 6). This in particular shows that for any fixed  $\ell$  the  $\ell$  least significant bits of the (binary representation of the) Hamming weight of the sticky random walk are nearly uniformly distributed.

► **Theorem 3.** *For any fixed  $m \geq 2$ , the total variation distance between the residues modulo  $m$  of the binomial random variable and the Hamming weight of the  $n$ -step sticky walk is at most  $\exp(-\Omega_m(n))$ .*

Recall that Ta-Shma established such a result for the  $m = 2$  case even for the expander random walk [12]. We execute a similar analysis, albeit only for the sticky walk, using  $m$ 'th roots of unity (in the place  $\pm 1$ ) to track the bias. We suspect extending this analysis to adjacency matrices of expanders (instead of the  $2 \times 2$  sticky walk transition matrix) can establish this equidistribution result for general expander walks, though we have not verified this.

Finally, the following confirms that analyzing sticky random walks is necessary in order to establish the corresponding claim for expander random walks. We should note that the graph family constructed does not fit the standard definition of an expander family, as degree-boundedness of the graph is not enforced. This is not a significant issue as the analysis of random walks on graphs always proceed by abstracting only the spectral properties of the graph. Further, we believe sampling a sparse regular subgraph with a similar structure should yield a similar claim for a bounded-degree expander family.

► **Theorem 4.** *There is a family of regular graphs whose nontrivial eigenvalues are bounded in magnitude by  $\lambda$ , half of whose vertices are marked, such that the bit string indicating which steps of a random walk visits a marked vertex has the same distribution as the sticky random walk.*

We conclude the introduction by mentioning a very interesting work of Bazzi on pseudobinominality [1]. This work establishes an upper bound on the total variation distance between the binomial distribution and the weight distribution of a  $\delta$ -biased random sequence, based

on the entropy of the latter distribution. One can show that the sticky random walk sequence  $\mathbf{s}$  is  $\lambda$ -biased (see Lemma 11). The bound on TVD in [1] is at best  $O(\sqrt{\delta n})$  and is only meaningful when the bias is tiny, which isn't the case for the sticky random walk. Also, calculating the entropy of the weight distribution of  $\mathbf{s}$  seems as hard, if not harder, than the Krawtchouk based calculations we use to directly bound the total variation distance in Section 4. In fact, it could be the case that the best approach to estimate the entropy of the weight distribution of the sticky random walk is via the entropy-difference bound [3] together with our bound on total variation distance.

## 1.2 Open Problems

Our results give rise to some immediate follow-up questions. Can the weighted  $\ell_2$  bound between the sticky and binomial distribution established in Lemma 13 be extended to give a non-trivial bound (that is bounded away from 1) for any fixed  $\lambda < 1$ ? Theorem 3 showed that any fixed number of least significant bits of (the Hamming weight of) the sticky walk are near uniform, but can this result be extended to other bits, for example the middle bit? Are there other symmetric properties of the sticky walk that resemble purely random strings?

A more important family of questions relate to extending our results from the sticky walk to the general expander walk model. Can the  $O(\lambda)$  TVD bound between the sticky and binomial distribution established in Theorem 1 be lifted to the general expander walk setting? Also, can the method of Krawtchouk functions used in Theorem 2 give insight towards unifying the Chernoff and parity bias results for expander random walks? In general, can we show distributions of various symmetric functions on expander walks are statistically close to the corresponding distributions on random strings? Moment generating function results from [7, 10] allow bounds on monotone symmetric functions to be lifted from the sticky walk to the expander walk, but no relationships are known for non-monotone functions like parity.

Recently, a TVD bound of  $O(\lambda \log^{3/2}(1/\lambda))$  between the Hamming weight of the general expander walk variant and the binomial distribution was proven using similar Fourier analytic techniques [2]. Removing this lower order polylogarithmic factor of  $\log^{3/2}(1/\lambda)$  remains an open problem. Theorem 4.8 in [2] also improved on Theorem 19 in this paper to an  $O(n^{-1/2})$  bound for the general expander walk, which consequently implies a tighter  $O(n^{-1/2})$  bound in the error term in Theorem 2. However, due to the lack of bit-flipping symmetry in the general expander walk, the extension of Theorem 2 to the expander walk remains open.

## 2 Preliminaries

### 2.1 Conventions and Notation

**Asymptotics.** In our asymptotic analysis, we will take  $\lambda$  to be constant and observe asymptotics as  $n \rightarrow \infty$ . Take  $o, O, \omega, \Omega$  to be the standard definitions. We will say  $f \lesssim g$  to mean  $f \leq Cg$  for some absolute constant  $C$  independent of  $n$  and  $\lambda$ . We also say  $f = g + O(h)$  when we mean  $|f - g| \leq O(h)$ , and analogously for  $o, \Omega$ , and  $\omega$ . We also denote  $\sim$  to be shorthand for  $= (1 + o(1))$ .

**Miscellaneous.** For a bit string  $s$ , we will denote  $|s|$  to be the Hamming weight of  $s$ .  $\mathcal{N}(\mu, \sigma^2)$  is the Gaussian distribution with mean  $\mu$  and variance  $\sigma^2$ . We write  $\text{Ber}(p)$  to be the Bernoulli distribution on  $\{0, 1\}$  where 1 has probability  $p$  and 0 has probability  $1 - p$ , and write  $\text{Bin}(n, 1/2)$  to be the binomial distribution of  $\sum_{i=1}^n b_i$  with independent choices of

$b_i \sim \text{Ber}(1/2)$ . Let  $\mathbf{1}_E$  represents the indicator variable of the event  $E$ . When written as a function,  $\mathbf{1}_S(i)$  is 1 if  $i \in S$ , and is 0 otherwise.  $[n] = \{1, 2, \dots, n\}$  denotes the set of the first  $n$  positive integers, and  $[n]_0 = \{0\} \cup [n]$ .  $\binom{[n]}{k}$  denotes the set of all size  $k$  subsets of  $[n]$ .

## 2.2 Definitions

In this paper, we will work with distributions on  $[n]_0$  and see how close they are to one another. For this reason, we state the standard definitions of the  $\ell_p$  distance between distributions

► **Definition 5** ( $\ell_p$  Distance/TVD). *Let  $Y$  and  $Z$  be distributions on  $[n]_0$ . For  $p \geq 1$ , we define the  $\ell_p$  distance between  $Y$  and  $Z$  to be*

$$\|Y - Z\|_p = \left( \sum_{i=0}^n |Y(i) - Z(i)|^p \right)^{1/p}.$$

The total variation distance (TVD) between  $Y$  and  $Z$  is simply  $\frac{1}{2}\|Y - Z\|_1$ .

Of course the first type of distribution we should formally define is the sticky random walk itself.

► **Definition 6** (Sticky Random Walk). *The sticky random walk  $S(n, \lambda)$  is a distribution on  $n$ -bit strings  $s$ , where  $s_1 \sim \text{Ber}(1/2)$ , and for  $2 \leq i \leq n$  and  $b \in \{0, 1\}$ , we have  $\Pr[s_i = b | s_{i-1} = b] = \frac{1+\lambda}{2}$ . In cases where  $n$  and  $\lambda$  are evident, only  $S$  may be used to denote the distribution.*

Define a bit string to be *homogeneous* if it only consists of 1's or of 0's. Given a bit string, we define a *run* to be a homogeneous substring that isn't a proper substring of another homogeneous substring. Intuitively, the sticky walk can be seen as a Markov chain on two states, where you stick to the same state with probability  $\frac{1+\lambda}{2}$  and transition to the other with probability  $\frac{1-\lambda}{2}$ . Thus the probability of a string is really only dependent on how many consecutive pair of bits are equal (equivalently the number of runs), rather than the precise value of the bits. Another thing to note is that  $\lambda$  can be seen as a parameter measuring the stickiness of a bit to its preceding one. Notice when  $\lambda = 0$  there is no stickiness present, and the sticky walk is just  $n$  independent coin flips.

## 2.3 Krawtchouk Functions

To analyze this sticky random walk, we heavily rely on the basis of Krawtchouk functions. Hence we define these functions here and state some standard identities of these functions without proof.

► **Definition 7** (Krawtchouk Functions). *The Krawtchouk function  $K_k : [n]_0 \rightarrow \mathbb{R}$  is defined to be*

$$K_k(\ell) = \sum_{\substack{y \in \{0,1\}^n \\ |y| = k}} (-1)^{\alpha \cdot y}$$

for each integer  $0 \leq \ell \leq n$  and an arbitrary  $n$ -bit string  $\alpha$  of Hamming weight  $\ell$  (the specific choice of  $\alpha$  does not matter due to symmetry).

## 48:6 Pseudobinomiality of the Sticky Random Walk

It can be shown that these functions form an orthogonal basis of the functions mapping  $[n]_0 \rightarrow \mathbb{R}$  with respect to the inner product

$$\langle f, g \rangle = \mathbb{E}_{\mathbf{b} \sim \text{Bin}(n, 1/2)} [f(\mathbf{b})g(\mathbf{b})]. \quad (1)$$

From the definition it is not hard to show that (e.g. Section 2.2 of [11]).

$$\mathbb{E}_{\mathbf{b} \sim \text{Bin}(n, 1/2)} [K_r(\mathbf{b})K_s(\mathbf{b})] = 0, \quad (2)$$

which shows that the Krawtchouk functions indeed form an orthogonal basis with respect to the inner product in (1). Furthermore, one can verify the following identities hold (see [11]).

$$\mathbb{E}_{\mathbf{b} \sim \text{Bin}(n, 1/2)} [K_k(\mathbf{b})^2] = \binom{n}{k} \quad (3)$$

$$\binom{n}{\ell} K_k(\ell) = \binom{n}{k} K_\ell(k). \quad (4)$$

Identities (2) and (3) allow us to explicitly expand any function uniquely as a sum of Krawtchouk functions.

► **Proposition 8.** *For function  $f : [n]_0 \rightarrow \mathbb{R}$ , there exists a unique expansion  $f(\ell) = \sum_{k=0}^n \hat{f}(k) K_k(\ell)$  where*

$$\hat{f}(k) = \frac{1}{\binom{n}{k}} \mathbb{E}_{\mathbf{b} \sim \text{Bin}(n, 1/2)} [f(\mathbf{b}) K_k(\mathbf{b})]$$

for each integer  $0 \leq k \leq n$ .

### 3 Using Krawtchouk Functions to Analyze the Sticky Walk

To help us analyze the sticky walk, we define the function  $p : [n]_0 \rightarrow \mathbb{R}$  to be  $p(\ell) = \frac{\Pr_{\mathbf{s} \sim S}[|\mathbf{s}|=\ell]}{\binom{n}{\ell} 2^{-n}}$ , and look at the Krawtchouk expansion. Doing so results in the following lemma.

► **Lemma 9.** *We have*

$$\hat{p}(k) = \frac{1}{\binom{n}{k}} \mathbb{E}_{\mathbf{s} \sim S} [K_k(|\mathbf{s}|)].$$

**Proof.** We just apply Proposition 8 to get that

$$\begin{aligned} \hat{p}(k) &= \frac{1}{\binom{n}{k}} \mathbb{E}_{\mathbf{b} \sim \text{Bin}(n, 1/2)} [p(\mathbf{b}) K_k(\mathbf{b})] \\ &= \frac{1}{\binom{n}{k}} \sum_{b=0}^n \binom{n}{b} 2^{-n} p(b) K_k(b) \\ &= \frac{1}{\binom{n}{k}} \sum_{b=0}^n \Pr_{\mathbf{s} \sim S} [|\mathbf{s}| = b] K_k(b) \\ &= \frac{1}{\binom{n}{k}} \mathbb{E}_{\mathbf{s} \sim S} [K_k(|\mathbf{s}|)]. \end{aligned} \quad \blacktriangleleft$$

What follows is a useful lemma that displays how Krawtchouk expanding  $p(\ell)$  can help analyze probabilities of the sticky distribution.



► **Lemma 10.** For  $\mathbf{s} \sim S(n, \lambda)$ , we can evaluate

$$\Pr[|\mathbf{s}| = \ell] = \frac{1}{2^n} \sum_{k=0}^n K_\ell(k) \mathbb{E}[K_k(|\mathbf{s}|)]$$

**Proof.** To estimate this probability, we can use Lemma 9 to find

$$\begin{aligned} \Pr[|\mathbf{s}| = \ell] &= \binom{n}{\ell} 2^{-n} p(\ell) \\ &= \frac{\binom{n}{\ell}}{2^n} \sum_{k=0}^n \hat{p}(k) K_k(\ell) \\ &= \frac{1}{2^n} \sum_{k=0}^n \frac{\binom{n}{\ell} K_k(\ell) \mathbb{E}[K_k(|\mathbf{s}|)]}{\binom{n}{k}} \\ &= \frac{1}{2^n} \sum_{k=0}^n K_\ell(k) \mathbb{E}[K_k(|\mathbf{s}|)] \end{aligned}$$

where we used the reciprocity relation (4) at the end. ◀

Notice that these lemmas didn't depend on the specific distribution  $S(n, \lambda)$ , and so these lemmas are applicable for arbitrary distributions on  $[n]_0$ . For our purposes, the sticky walk versions will be used for our analysis done in the next sections.

## 4 Total Variation Distance Bounds

### 4.1 Upper Bounding the TVD Between the Sticky and Binomial Distribution

Upper bounding the TVD requires calculating the expectation of various sums and products of sticky walk random variables. We will abstract out these calculations in the following lemmas. Analogous expressions for the general expander walk model were also analyzed in Lemmas 3.3 and 4.2 in Rao and Regev [10]. In particular, [10] gives upper bounds on these quantities in the general expander walk, while we give exact values for the simpler sticky walk.

► **Lemma 11.** Let  $\mathbf{s} \sim S(n, \lambda)$ . For even-sized subsets  $A \subset [n]$  where  $a_1 < \dots < a_m$  are the elements of  $A$  in increasing order, define  $\text{shift}(A) = \sum_{i=1}^{|A|/2} (a_{2i} - a_{2i-1})$ . For any  $A \subset [n]$ , we have

$$\mathbb{E} \left[ \prod_{i \in A} (-1)^{\mathbf{s}_i} \right] = \begin{cases} 0 & |A| \text{ odd} \\ \lambda^{\text{shift}(A)} & |A| \text{ even} \end{cases}.$$

**Proof.** Since the sticky walk is a Markov chain where  $(-1)^{\mathbf{s}_i}$  has the same sign as  $(-1)^{\mathbf{s}_{i-1}}$  with probability  $\frac{1+\lambda}{2}$ , we can rewrite these random variables in terms of a product of independent random variables representing the transitions of the chain. In particular, we define a new random variable  $\mathbf{u} \in \{0, 1\}^n$ , where  $\mathbf{u}_1 \sim \text{Ber}(1/2)$  and  $\mathbf{u}_i \sim \text{Ber}(\frac{1-\lambda}{2})$  for  $2 \leq i \leq n$ . One can easily check that  $(-1)^{\mathbf{s}_i}$  is the same random variable as  $\prod_{j=1}^i (-1)^{\mathbf{u}_j} = (-1)^{\sum_{j=1}^i \mathbf{u}_j}$ . Hence we have

$$\mathbb{E} \left[ \prod_{i \in A} (-1)^{\mathbf{s}_i} \right] = \mathbb{E} [ (-1)^{\sum_{i \in A} \sum_{j=1}^i \mathbf{u}_j} ] = \prod_{j=1}^{a_m} \mathbb{E} [ (-1)^{\sum_{i \in A; i \geq j} \mathbf{u}_j} ]$$

Note when  $|A|$  is odd, the factor when  $j = 1$  is 0 since  $\mathbb{E}[(-1)^{\sum_{i \in A} u_i}] = \mathbb{E}[(-1)^{u_1}] = 0$ . Hence the total expectation is 0. Otherwise, when  $|A|$  is even, the  $j = 1$  term is just  $\mathbb{E}[(-1)^{|A|u_1}] = 1$ . For  $j \geq 2$ , one sees that if  $A_j = \{i \in A : i \geq j\}$  is of odd cardinality, then  $\mathbb{E}[(-1)^{\sum_{i \in A; i \geq j} u_i}] = \mathbb{E}[(-1)^{u_j}] = \lambda$ , and is 1 otherwise. The set of  $j$  such that  $|A_j|$  is odd is simply the integers in  $(a_1, a_2] \cup (a_3, a_4] \cup \dots \cup (a_{m-1}, a_m]$ , of which there are  $\text{shift}(A)$ . Consequently, upon multiplying all the  $j$  factors, we derive that the expectation is  $\lambda^{\text{shift}(A)}$  for  $|A|$  even.  $\blacktriangleleft$

► **Lemma 12.** For all nonnegative integers  $k$  and  $s \sim S(n, \lambda)$ ,

- $\mathbb{E}[K_{2k}(|s|)] = \sum_{m=k}^{n-k} \binom{m-1}{k-1} \binom{n-m}{k} \lambda^m$ , and
- $\mathbb{E}[K_{2k+1}(|s|)] = 0$

with the convention that  $\binom{-1}{-1} = 1$  and  $\binom{j}{-1} = 0$  for  $j \geq 0$ .

**Proof.** Using the definition, we rewrite

$$K_k(|s|) = \sum_{\substack{\alpha \in \{0,1\}^n \\ |\alpha| = k}} (-1)^{\sum_{i=1}^n \alpha_i s_i} = \sum_{T \in \binom{[n]}{k}} (-1)^{\sum_{i \in T} s_i} = \sum_{T \in \binom{[n]}{k}} \prod_{i \in T} (-1)^{s_i}. \quad (5)$$

Then from Lemma 11, we have

$$\mathbb{E}[K_k(|s|)] = \sum_{T \in \binom{[n]}{k}} \mathbb{E} \left[ \prod_{i \in T} (-1)^{s_i} \right] = \begin{cases} 0 & k \text{ odd} \\ \sum_{T \in \binom{[n]}{k}} \lambda^{\text{shift}(T)} & k \text{ even} \end{cases}.$$

We now evaluate

$$\mathbb{E}[K_{2k}(|s|)] = \sum_{T \in \binom{[n]}{2k}} \lambda^{\text{shift}(T)} = \sum_{m=k}^{n-k} \left( \sum_{\substack{T \in \binom{[n]}{2k} \\ \text{shift}(T) = m}} 1 \right) \lambda^m$$

Hence to prove the lemma, it suffices to show the number of subsets  $T \in \binom{[n]}{2k}$  with  $\text{shift}(T) = m$  is  $\binom{m-1}{k-1} \binom{n-m}{k}$ . Let  $t_1 < t_2 < \dots < t_{2k}$  be the elements of  $T$ . We claim the desired subsets  $T$  are in bijection with a  $k$ -tuple  $(d_1, \dots, d_k)$  of positive integers summing to  $m$ , paired with an  $n - m$  letter word consisting of  $n - m - k$   $A$ 's and  $k$   $B$ 's. To construct the pair from a subset  $T$ , we can set  $d_i = t_{2i} - t_{2i-1}$  and the  $n - m$  letter word to be  $A^{t_1-1} C B A^{n-t_{2k}}$  where  $C = \bigcirc_{i=1}^{k-1} (B A^{t_{2i+1}-t_{2i}-1})$  ( $\bigcirc$  is concatenation). Since  $\text{shift}(T) = m$ , the  $d_i$  are positive integers summing to  $m$ , and the word can easily be verified to have  $n - m - k$   $A$ 's and  $k$   $B$ 's.

Now given a  $(d_1, \dots, d_k)$  and a word, we can construct a  $T$  with  $\text{shift}(T) = m$  as follows. The word will be of the form  $A^{a_1} B A^{a_2} B \dots A^{a_k} B A^{a_{k+1}}$  where each  $a_i \geq 0$ . We then set  $t_1 = a_1 + 1$ , and assign  $t_2, \dots, t_{2k}$  inductively as follows.

$$t_{2i} = t_{2i-1} + d_i$$

$$t_{2i+1} = t_{2i} + a_{i+1} + 1$$

Finally we set  $T = \{t_i\}_{i=1}^{2k}$ . It can then be verified all  $t_i \in [n]$  and  $\text{shift}(T) = m$ . In fact, one can check the two constructed maps are inverses of each other. Hence we have established the bijection and counting the number of such  $T$  is simply counting the number of  $k$ -tuples and word pairs, which by standard counting methods is  $\binom{m-1}{k-1} \binom{n-m}{k}$ . The desired result follows.  $\blacktriangleleft$

We are now ready to establish some upper bounds using the lemmas above. We first upper bound the weighted  $\ell_2$  distance between the sticky walk and binomial distribution by defining a suitable function and taking the Krawtchouk expansion.

► **Lemma 13.** *Let  $\mathbf{s} \sim S(n, \lambda)$  for  $\lambda < .16$  and define  $p(\ell) := \frac{\Pr_{\mathbf{s} \sim S}(|\mathbf{s}| = \ell)}{\binom{n}{\ell} 2^{-n}}$ . Then we have*

$$\mathbb{E}_{\mathbf{b} \sim \text{Bin}(n, 1/2)} [(p(\mathbf{b}) - 1)^2] \leq O(\lambda^2).$$

**Proof.** Let us write  $p(\ell) = \sum_{k=0}^n \hat{p}(k) K_k(\ell)$  where

$$\hat{p}(k) = \frac{1}{\binom{n}{k}} \mathbb{E}_{\mathbf{b} \sim \text{Bin}(n, 1/2)} [p(\mathbf{b}) K_k(\mathbf{b})] \quad (6)$$

from Proposition 8. From the definition, we can verify  $K_0(\ell) = 1$  for all  $\ell$ . Combining this fact with (6) implies that  $\hat{p}(0) = 1$  too. Hence,

$$\begin{aligned} \mathbb{E}_{\mathbf{b} \sim \text{Bin}(n, 1/2)} [(p(\mathbf{b}) - 1)^2] &= \mathbb{E}_{\mathbf{b} \sim \text{Bin}(n, 1/2)} \left[ \left( \sum_{k=1}^n \hat{p}(k) K_k(\mathbf{b}) \right)^2 \right] \\ &= \sum_{k=1}^n \hat{p}(k)^2 \binom{n}{k} \\ &= \sum_{k=1}^n \frac{1}{\binom{n}{k}} \mathbb{E}[K_k(|\mathbf{s}|)]^2 \end{aligned} \quad (7)$$

where only the diagonal terms of the square survive due to the orthogonality relations (2) and (3). Adding the  $k = 0$  term back in (7) will simply give the expression for  $\mathbb{E}[p(\mathbf{b})^2]$  (and will be stated as a subsequent corollary).

From Lemma 12 and the generating function relation  $(\frac{x}{1-x})^k = \sum_{m \geq k} \binom{m-1}{k-1} x^m$ ,

$$\mathbb{E}_{\mathbf{b} \sim \text{Bin}(n, 1/2)} [(p(\mathbf{b}) - 1)^2] = \sum_{1 \leq k \leq n/2} \frac{1}{\binom{n}{2k}} \left( \sum_{m=k}^{n-k} \binom{m-1}{k-1} \binom{n-m}{k} \lambda^m \right)^2 \quad (8)$$

$$\begin{aligned} &\leq \sum_{1 \leq k \leq n/2} \frac{\binom{n}{k}^2}{\binom{n}{2k}} \left( \sum_{m=k}^{n-k} \binom{m-1}{k-1} \lambda^m \right)^2 \\ &\leq \sum_{1 \leq k \leq n/2} \frac{\binom{n}{k}^2}{\binom{n}{2k}} \left( \frac{\lambda}{1-\lambda} \right)^{2k} \end{aligned} \quad (9)$$

With the following claim (whose proof will be deferred to the appendix) we can deduce a lower bound.

► **Claim 14.** For  $1 \leq k \leq n/2$ , we can bound  $\frac{\binom{n}{k}^2}{\binom{n}{2k}} \leq \frac{e^3 \sqrt{2}}{4\pi^2} \cdot 4^{2k}$ .

Hence by combining (9) and Claim 14, we conclude

$$\begin{aligned} \mathbb{E}_{\mathbf{b} \sim \text{Bin}(n, 1/2)} [(p(\mathbf{b}) - 1)^2] &\leq \frac{e^3 \sqrt{2}}{4\pi^2} \sum_{1 \leq k \leq n/2} \left( \frac{4\lambda}{1-\lambda} \right)^{2k} \\ &\leq \frac{4e^3 \sqrt{2}}{\pi^2} \frac{\lambda^2}{(1-5\lambda)(1+3\lambda)} \\ &< \left( \frac{\lambda}{0.16} \right)^2, \end{aligned}$$

which is a nontrivial  $O(\lambda^2)$  bound that is strictly less than 1. ◀

## 48:10 Pseudobinomiality of the Sticky Random Walk

Remembering our prior observation that the sum in Equation (7) is 1 less than  $\mathbb{E}[p(\mathbf{b})^2]$ , we immediately get the following corollary.

► **Corollary 15.** *Let  $\mathbf{s}, p$  be defined as in Lemma 13. We can then bound*

$$\mathbb{E}_{\mathbf{b} \sim \text{Bin}(n, 1/2)}[p(\mathbf{b})^2] = \sum_{k=0}^n \frac{1}{\binom{n}{k}} \mathbb{E}[K_k(|\mathbf{s}|)]^2 \leq 1 + O(\lambda^2).$$

Going back to proving the main result of the section, the brunt of the work was actually done in Lemma 13. We can now simply apply convexity to establish the desired TVD upper bound between the sticky and binomial distribution.

► **Theorem 16.** *Let  $\mathbf{s} \sim S(n, \lambda)$ . We can then bound the TVD between  $S(n, \lambda)$  and  $\text{Bin}(n, 1/2)$ ,*

$$\frac{1}{2} \sum_{\ell=0}^n \left| \Pr[|\mathbf{s}| = \ell] - \binom{n}{\ell} 2^{-n} \right| \leq O(\lambda).$$

**Proof.** By convexity, and then Lemma 13, we get

$$\sum_{\ell=0}^n \left| \Pr[|\mathbf{s}| = \ell] - \binom{n}{\ell} 2^{-n} \right| = \mathbb{E}_{\mathbf{b} \sim \text{Bin}(n, 1/2)}[|p(\mathbf{b}) - 1|] \leq \sqrt{\mathbb{E}_{\mathbf{b} \sim \text{Bin}(n, 1/2)}[(p(\mathbf{b}) - 1)^2]} \leq O(\lambda)$$

for  $\lambda < 0.16$ , and clearly the TVD is bounded by 1 for  $\lambda$  greater. ◀

### 4.2 Limitations to the Upper Bound Approach

Unfortunately, the weighted  $\ell_2$  distance studied in Lemma 13 blows up as  $\lambda$  approaches 1. Looking at the term in the sum of Equation (8) when  $k = m = (1/2 - \varepsilon)n$  for some  $\varepsilon > 0$  (will be specified later), and applying the bounds  $\left(\frac{n}{k}\right)^k \leq \binom{n}{k} \leq \left(\frac{n\varepsilon}{k}\right)^k$  yields

$$\begin{aligned} \sum_{1 \leq k \leq n/2} \frac{1}{\binom{n}{2k}} \left( \sum_{m=k}^{n-k} \binom{m-1}{k-1} \binom{n-m}{k} \lambda^m \right)^2 &\geq \frac{1}{\binom{n}{(1-2\varepsilon)n}} \left( \frac{(1/2 + \varepsilon)n}{(1/2 - \varepsilon)n} \right)^2 \lambda^{(1/2 - \varepsilon)n} \\ &\geq \left( \frac{n}{2\varepsilon n} \right)^{-1} \left( \frac{(1/2 + \varepsilon)n}{2\varepsilon n} \right)^2 \lambda^{n/2} \\ &\geq \left( \frac{\lambda^{1/2}}{(8\varepsilon)^{2\varepsilon}} \right)^n \end{aligned}$$

For  $\lambda \geq .94$  and  $\varepsilon = .017$ , we have  $\lambda^{1/2} > .969$  and  $(8\varepsilon)^{2\varepsilon} < .967$ . Thus

$$\mathbb{E}_{\mathbf{b} \sim \text{Bin}(n, 1/2)}[(p(\mathbf{b}) - 1)^2] > \left( \frac{.969}{.967} \right)^n$$

which grows exponentially in  $n$ . Consequently, in order to show an  $O(\lambda)$  bound for all  $\lambda < 1$ , an approach different from using the weighted  $\ell_2$  distance is required.

### 4.3 Showing the TVD Bound is Tight

Consider  $\mathbf{s} \sim S(n, \lambda)$  and  $\mathbf{N} \sim \mathcal{N}(0, 1)$ . Set  $\mathbf{Y}_i = (-1)^{\mathbf{s}_i}$  and let  $\mathbf{Y} = \sum_{i=1}^n (-1)^{\mathbf{s}_i}$  be the  $\pm 1$  variant of the sticky distribution. Similarly, let  $b_i \sim \text{Ber}(1/2)$  for  $1 \leq i \leq n$ . Set  $\mathbf{Z}_i = (-1)^{b_i}$  and let  $\mathbf{Z} = \sum_{i=1}^n \mathbf{Z}_i$  be the usual  $\pm 1$  unbiased random walk. By the central limit theorem, we already know  $\frac{\mathbf{Z}}{\sqrt{n}} \rightarrow \mathbf{N}$  in distribution as  $n \rightarrow \infty$ . We now state a similar result for  $\mathbf{Y}$ , and defer the proof to the appendix, as it requires lengthy calculation of the moments.

► **Lemma 17.** *As  $n \rightarrow \infty$ ,  $\mathbf{Y} \sqrt{\frac{1-\lambda}{(1+\lambda)n}} \rightarrow \mathcal{N}$  in distribution.*

With this lemma, we have a good understanding of  $\mathbf{Y}$ , and can show the tightness of the TVD bound. The suggestion to consider an event like  $|\mathbf{Y}| \leq O(\sqrt{n})$  was made by Salil Vadhan [13].

► **Theorem 18.** *For  $\mathbf{Y}$  and  $\mathbf{Z}$  defined above as the  $\pm 1$  version of the  $S(n, \lambda)$  walk and the  $n$ -step unbiased walk, respectively, we have*

$$|\Pr[|\mathbf{Z}| \leq \sqrt{n}] - \Pr[|\mathbf{Y}| \leq \sqrt{n}]| \geq \Omega(\lambda)$$

for  $\lambda < 1$ .

**Proof.** By Lemma 17, for  $\varepsilon < \int_{\sqrt{\frac{1-\lambda}{1+\lambda}}}^1 e^{-x^2} dx$ , there exists a constant  $N(\varepsilon)$  such that for  $n > N(\varepsilon)$ ,

$$\begin{aligned} |\Pr[|\mathbf{Z}| \leq \sqrt{n}] - \Pr[|\mathbf{Y}| \leq \sqrt{n}]| &= \left| \Pr\left[\frac{|\mathbf{Z}|}{\sqrt{n}} \leq 1\right] - \Pr\left[|\mathbf{Y}| \sqrt{\frac{1-\lambda}{(1+\lambda)n}} \leq \sqrt{\frac{1-\lambda}{1+\lambda}}\right] \right| \\ &\geq 2 \int_{\sqrt{\frac{1-\lambda}{1+\lambda}}}^1 e^{-x^2} dx - \varepsilon \\ &\geq \int_{\sqrt{\frac{1-\lambda}{1+\lambda}}}^1 e^{-x^2} dx \geq e^{-1} \left(1 - \sqrt{\frac{1-\lambda}{1+\lambda}}\right) \geq \frac{\lambda}{2e} \end{aligned}$$

where the last step follows from the easily verifiable inequality  $1 - \sqrt{\frac{1-x}{1+x}} \geq \frac{x}{2}$  for  $0 \leq x \leq 1$ . Hence we have demonstrated an event that gives an  $\Omega(\lambda)$  gap between the distributions of  $\mathbf{Y}$  and  $\mathbf{Z}$  for  $\lambda < 1$ . Since  $\mathbf{Y}$  and  $\mathbf{Z}$  are just shifted and dilated versions of the Hamming weight of  $S(n, \lambda)$  and  $\text{Bin}(n, 1/2)$ , respectively, we can deduce the TVD is  $\Theta(\lambda)$  due to Theorem 16. ◀

## 5 Parity and Majority of the Sticky Walk are Almost Uncorrelated

Following the spirit of how Ta-Shma [12] showed the bias of the parity of an expander walk sampler is exponentially small, we show some parity events of the sticky random walk are very close in probability to the corresponding probability under the binomial distribution. An interesting question to consider is whether more refined events, such as whether the output is even and above expectation is close to  $\frac{1}{4}$ . We show such results for the sticky distribution. Note that from Theorem 16 we know the event probability in the sticky distribution will be within  $O(\lambda)$  of  $\frac{1}{4}$ , but in this section, we derive an  $o(1)$  error bound.

Let  $\mathbf{s} \sim S(n, \lambda)$ . We first demonstrate that similar to the binomial distribution,  $\Pr[|\mathbf{s}| \text{ even}]$  and  $\Pr[|\mathbf{s}| \geq n/2]$  are close to  $1/2$ . For the parity, we straightforwardly calculate

$$\Pr[|\mathbf{s}| \text{ even}] - \Pr[|\mathbf{s}| \text{ odd}] = \sum_{t=0}^n (-1)^t \Pr[|\mathbf{s}| = t] = \mathbb{E}[(-1)^{|\mathbf{s}|}] = \lambda^{n/2} \cdot \mathbf{1}_{n \text{ even}} \quad (10)$$

using Lemma 11. Hence  $|\Pr[|\mathbf{s}| \text{ even}] - \frac{1}{2}| \leq \frac{\lambda^{n/2}}{2}$ . Interestingly enough, this  $\lambda^{n/2}$  bias is also present in the parity bias calculation done by Ta-Shma for the expander walk in [12] (Section 3.2). Furthermore, this calculation (combined with our work in Section 7) shows that this error term cannot be improved to something smaller like  $\lambda^n$ .

## 48:12 Pseudobinomiality of the Sticky Random Walk

Let  $a$  be a string and let  $\bar{a}$  be the string formed by toggling every bit of  $a$ . Notice that  $\Pr[\mathbf{s} = a] = \Pr[\mathbf{s} = \bar{a}]$  since the number of runs are the same in both strings. Summing over all strings of Hamming weight  $t$ , we get the symmetric relation  $\Pr[|\mathbf{s}| = t] = \Pr[|\mathbf{s}| = n - t]$ , which directly implies  $\Pr[|\mathbf{s}| > n/2] = \Pr[|\mathbf{s}| < n/2]$ . In the case  $n$  is odd, we immediately have  $\Pr[|\mathbf{s}| > n/2] = 1/2$ . When  $n$  is even, we have

$$\Pr[|\mathbf{s}| > n/2] = \frac{1}{2} - \frac{\Pr[|\mathbf{s}| = n/2]}{2}. \quad (11)$$

We now prove a lemma that will help us bound  $\Pr[|\mathbf{s}| = n/2]$  in the sticky distribution.

► **Theorem 19.** *For  $\mathbf{s} \sim S(n, \lambda)$ ,  $\lambda < 1/5$ , and  $\mathbf{b} \sim \text{Bin}(n, 1/2)$ , we can bound*

$$\Pr[|\mathbf{s}| = \ell] \lesssim \sqrt{(1 + O(\lambda^2)) \Pr[\mathbf{b} = \ell]}.$$

**Proof.** To estimate this probability, we can use Lemma 10 to find

$$\Pr[|\mathbf{s}| = \ell] = \frac{1}{2^n} \sum_{k=0}^n K_\ell(k) \mathbb{E}[K_k(|\mathbf{s}|)] \leq \frac{1}{2^n} \sqrt{\sum_{k=0}^n \binom{n}{k} K_\ell(k)^2} \sqrt{\sum_{k=0}^n \frac{\mathbb{E}[K_k(|\mathbf{s}|)]^2}{\binom{n}{k}}} \quad (12)$$

by Cauchy-Schwarz. Notice by Corollary 15,

$$\sqrt{\sum_{k=0}^n \frac{\mathbb{E}[K_k(|\mathbf{s}|)]^2}{\binom{n}{k}}} \lesssim \sqrt{1 + O(\lambda^2)}. \quad (13)$$

Finally, by using Equation (3) we get

$$\sqrt{\sum_{k=0}^n \binom{n}{k} K_\ell(k)^2} = \sqrt{\binom{n}{\ell} 2^n} \quad (14)$$

Combining (12), (13), and (14) yields

$$\Pr[|\mathbf{s}| = \ell] \lesssim \frac{1}{2^n} \sqrt{\binom{n}{\ell} 2^n (1 + O(\lambda^2))} = \sqrt{(1 + O(\lambda^2)) \Pr[\mathbf{b} = \ell]}$$

as desired. ◀

Going back to (11), Theorem 19 with  $\ell = n/2$  now allows us to deduce

$$\Pr[|\mathbf{s}| > n/2] = \frac{1}{2} + O(n^{-1/4} \sqrt{1 + \lambda^2}).$$

Thus, we have shown that  $\Pr[|\mathbf{s}| \text{ even}]$  and  $\Pr[|\mathbf{s}| > n/2]$  are near  $1/2$ , which are properties shared by purely random strings. However, we can go further and show a more refined equidistribution result. One can calculate for  $\mathbf{b} \sim \text{Bin}(n, 1/2)$  that  $\Pr[(|\mathbf{b}| \text{ odd}) \wedge (|\mathbf{b}| > n/2)] = \frac{1}{4} + O(n^{-1/2})$ . We show an analogous result for the sticky walk, albeit with a worse  $o(1)$  error term.

► **Theorem 20.** *Let  $\mathbf{s} \sim S(n, \lambda)$ . For  $a, b \in \{0, 1\}$ , denote the event*

$$E_{ab} = (|\mathbf{s}| \equiv a \pmod{2}) \wedge ((-1)^b |\mathbf{s}| > (-1)^b n/2)$$

and  $p_{ab} = \Pr[E_{ab}]$ . Then  $p_{ab} = \frac{1}{4} + O(n^{-1/4} \sqrt{1 + \lambda^2})$  for all  $a, b$ .

**Proof.** We show the result for  $p_{00}$  (symmetric arguments work for any  $p_{ab}$ ). We split into cases when  $n$  is even and odd. When  $n$  is even, notice that due to the symmetry  $\Pr[|s| = t] = \Pr[|s| = n - t]$ ,  $p_{00} = p_{01}$  and  $p_{10} = p_{11}$ . Hence from (10) we have  $\lambda^{n/2} \geq |p_{10} + p_{11} - p_{01} - p_{00}| = 2|p_{10} - p_{00}|$ . From (11) and Theorem 19, we have  $p_{10} + p_{00} = \frac{1}{2} + O(n^{-1/4}\sqrt{1+\lambda^2})$ , and so we can conclude  $p_{00} = \frac{1}{4} + O(n^{-1/4}\sqrt{1+\lambda^2})$  as desired.

When  $n$  is odd, a little more effort is required to exploit symmetry. Let  $S$  be the set of  $n$ -bit strings having a run of size  $\geq 2$  which doesn't contain the  $n$ th bit, and let  $T$  be the set of  $n$ -bit strings having a run of size  $\geq 2$  which doesn't contain the 1st bit. Consider the map  $g : S \rightarrow T$  defined by toggling the last bit of the first run of size  $\geq 2$ . It can be easily seen that  $g$  is a bijection. Note that for any  $s \in S$ ,  $s$  and  $g(s)$  have the same probability under the sticky walk distribution (the map preserves the number of runs in the string). Furthermore,  $g$  changes the parity of the input string. The image of all strings in  $S$  with even parity and Hamming weight  $> n/2$  under  $g$  will be all strings in  $T$  with odd parity and Hamming weight  $> n/2$ , along with some rogue strings in  $T$  with Hamming weight  $(n-1)/2$ . If we isolate these rogue cases, we can deduce, due to the fact  $g$  preserves probability, that

$$\begin{aligned} \Pr[(s \in S) \wedge E_{00}] &= \Pr[(s \in T) \wedge E_{10}] + \Pr\left[(s \in S) \wedge \left(|g(s)| = \frac{n-1}{2}\right) \wedge E_{00}\right] \\ &\leq \Pr[(s \in T) \wedge E_{10}] + O(n^{-1/4}\sqrt{1+\lambda^2}) \end{aligned} \quad (15)$$

by Theorem 19 with  $\ell = \frac{n-1}{2}$ . Notice any  $n$ -bit string has probability measure at most  $\frac{1}{2} \left(\frac{1+\lambda}{2}\right)^{n-1}$  in the sticky distribution. Furthermore, there are only  $O(n)$  strings not contained in  $S$  (such strings must be of form  $0^k A$  or  $1^k A$  where  $A$  is a binary string that alternates 0s and 1s). Therefore we can bound

$$|\Pr[(s \in S) \wedge E_{00}] - \Pr[E_{00}]| \leq \Pr[s \notin S] \lesssim n \left(\frac{1+\lambda}{2}\right)^n. \quad (16)$$

An analogous argument with set  $T$  gives

$$|\Pr[(s \in T) \wedge E_{01}] - \Pr[E_{01}]| \lesssim n \left(\frac{1+\lambda}{2}\right)^n. \quad (17)$$

Combining (15), (16), and (17) using an application of the triangle inequality allows us to bound

$$\begin{aligned} |p_{00} - p_{01}| &\leq |\Pr[E_{00}] - \Pr[(s \in S) \wedge E_{00}]| + |\Pr[(s \in S) \wedge E_{00}] - \Pr[(s \in T) \wedge E_{01}]| \\ &\quad + |\Pr[(s \in T) \wedge E_{01}] - \Pr[E_{01}]| \\ &\lesssim n^{-1/4}\sqrt{1+\lambda^2}. \end{aligned}$$

Since  $p_{00} + p_{01} = 1/2$  for  $n$  odd, we can deduce  $p_{00} = \frac{1}{4} + O(n^{-1/4}\sqrt{1+\lambda^2})$ . ◀

## 6 Sticky Walk Modulo $m$ is Close to Uniform

In [12], Ta-Shma provides an argument on how the parity of the expander walk is unbiased. In this section we use his method to show the sticky distribution modulo  $m$  is approximately uniform for any fixed  $m$  and large  $n$ .

► **Lemma 21.** *Fix  $\lambda < 1$  and  $m \geq 2$ . For any nontrivial  $m$ 'th root of unity  $\zeta \neq 1$  and  $s \sim S(n, \lambda)$ , we have  $|\mathbb{E}[\zeta^{|s|}]| \leq \exp(-\Omega(n))$ .*



## 48:14 Pseudobinomiality of the Sticky Random Walk

**Proof.** One can verify  $E[\zeta^{|s|}] = \mathbf{1}^\top M^n \mathbf{1}$  where  $\mathbf{1}$  is the unit vector with all coordinates equal and

$$M = \begin{pmatrix} \frac{1+\lambda}{2} & \frac{1-\lambda}{2} \\ \frac{1-\lambda}{2} & \frac{1+\lambda}{2} \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & \zeta \end{pmatrix} = \begin{pmatrix} \frac{1+\lambda}{2} & \frac{1-\lambda}{2}\zeta \\ \frac{1-\lambda}{2} & \frac{1+\lambda}{2}\zeta \end{pmatrix}.$$

Let  $\lambda_1, \lambda_2$  be the eigenvalues of  $M$ . If  $M$  is diagonalizable, we know  $\mathbf{1}^\top M^n \mathbf{1} = c_1 \lambda_1^n + c_2 \lambda_2^n$ , and if  $M$  is not diagonalizable we can write  $\mathbf{1}^\top M^n \mathbf{1} = (c_1 + nc_2) \lambda_1^n$  for constants  $c_1$  and  $c_2$  (this can be seen by writing  $M$  in Jordan normal form). Thus in either case, it suffices to show the norm of both eigenvalues are strictly less than 1 to prove the lemma. From Gershgorin's circle theorem, we know the norms of the eigenvalues are at most  $\frac{1+\lambda}{2} + \frac{1-\lambda}{2} = 1$ .

Assume one eigenvalue has norm 1. Since the product of the eigenvalues is  $\det(M) = \lambda\zeta$ , we deduce the eigenvalues are  $\lambda z, \zeta/z$  for some  $|z| = 1$ . Since  $\lambda z + \zeta/z = \text{Tr}(M) = \frac{1+\lambda}{2}(1+\zeta)$ , we particularly know  $\lambda z + \zeta/z$  and  $1 + \zeta$  have the same argument angle.

If  $\zeta \neq -1$ , we must have the quotient of these two complex numbers be real. Consequently

$$0 = \frac{\lambda z + \zeta/z}{1 + \zeta} - \overline{\left( \frac{\lambda z + \zeta/z}{1 + \zeta} \right)} = \frac{\lambda z + \zeta/z}{1 + \zeta} - \frac{\lambda/z + z/\zeta}{1 + 1/\zeta} = \frac{(\lambda - 1)(z - \zeta/z)}{1 + \zeta}.$$

Since  $\lambda \neq 1$ , we have  $z = \zeta/z$  and so the eigenvalues are  $\lambda z, z$ . The trace is  $\frac{1}{2}(1 + \lambda)(1 + \zeta) = \frac{1}{2}(1 + \lambda)(1 + z^2)$ , so

$$\lambda z + z = \frac{1}{2}(1 + \lambda)(1 + z^2) \iff z = \frac{1 + z^2}{2} \iff z = 1.$$

However  $\zeta = z^2 = 1$ , a contradiction.

If  $\zeta = -1$ , then  $\lambda z, -1/z$  are the eigenvalues. The trace is now zero, so  $\lambda z = 1/z$ . Since  $\lambda$  is a nonnegative real, taking norms of both sides implies  $\lambda = 1$ , a contradiction. Hence the eigenvalues are indeed less than 1.  $\blacktriangleleft$

Lemma 21 allows us to employ Lemma 4.2 in [9] (since all characters of  $\mathbb{Z}/m\mathbb{Z}$  are  $\chi(n) = \zeta^n$  for some  $m$ 'th root of unity  $\zeta$ ) to deduce the following.

► **Theorem 22.** *Let  $U_m$  be the uniform distribution over  $[m]$ , and let  $S_m(n, \lambda)$  be the distribution of  $|s| \pmod{m}$  over  $[m]$ , where  $s \sim S(n, \lambda)$ . The  $\ell_1$  distance between these two distributions can then be bounded by*

$$\|U_m - S_m(n, \lambda)\|_1 \leq \exp(-\Omega(n))$$

where the implied constants only depend on  $m$ .

## 7 Relationship Between the Sticky Walk and Expander Walk

Define a  $\lambda$ -expander<sup>1</sup> to be a regular graph  $G$  such that all eigenvalues  $\lambda_1 \leq \dots \leq \lambda_m$  of the normalized adjacency matrix satisfy  $|\lambda_i| \leq \lambda$  for  $1 \leq i \leq m - 1$ . One of the main motivations to study the sticky random walk is because of its perceived close relationship with the distributions generated by expander walks. In particular, if  $(v_1, \dots, v_n)$  is a  $n$ -step

<sup>1</sup> Unlike standard definitions, we do not restrict the degree of each vertex to be constant. The purpose of this section is to show any analysis of expander walks cannot give better bounds than the sticky walk. Since analyses on expander walks are based off of the spectral properties of the graph rather than its degree, restriction of the degrees are unnecessary for our purposes.

expander walk on a  $\lambda$ -expander  $G = (V, E)$ , and  $W \subset V$  with  $|W| = |V|/2$ , we believe the distribution  $(\mathbf{1}_{v_1 \in W}, \dots, \mathbf{1}_{v_n \in W})$  is linked to the sticky walk  $S(n, \lambda)$ . In this section, we demonstrate one direction of this relationship by explicitly constructing a  $\lambda$ -expander and vertex subset  $W$  of half the size such that the distribution of  $(\mathbf{1}_{v_1 \in W}, \dots, \mathbf{1}_{v_n \in W})$  is precisely  $S(n, \lambda)$ .

► **Theorem 23.** *There exists  $\lambda$ -expander  $G = (V, E)$  and vertex set  $W \subset V$  with  $|W| = |V|/2$  such that if  $(v_1, \dots, v_n)$  is a random walk on  $G$ , the random  $n$ -bit string  $(\mathbf{1}_{v_1 \in W}, \dots, \mathbf{1}_{v_n \in W}) \sim S(n, \lambda)$ .*

**Proof.** For simplicity, assume  $\lambda$  is rational, and take integer  $m$  such that  $(\frac{1-\lambda}{1+\lambda})m$  is an integer. Let  $C_0$  and  $C_1$  be two  $m$ -cliques with an added self-loop at each vertex. Construct the graph  $G$  by taking each vertex in  $C_0$  and connect it to  $(\frac{1-\lambda}{1+\lambda})m$  vertices in  $C_1$  in a cyclic uniform manner to make this graph  $\frac{2m}{1+\lambda}$ -regular (i.e. if we arbitrarily number the vertices in  $C_0$  and  $C_1$  from 1 to  $m$ , just connect vertex  $i$  in  $C_0$  with the  $C_1$  vertices  $i, i+1, \dots, i + (\frac{1-\lambda}{1+\lambda})m - 1 \pmod{m}$ ). Note upon setting  $W = C_1$ , a random walk on this expander resembles the sticky random walk, because at each step,  $m$  edges will keep us in the same clique, and  $(\frac{1-\lambda}{1+\lambda})m$  will move us to the other, which gives us a  $\frac{1+\lambda}{2}$  chance of staying in the same clique and  $\frac{1-\lambda}{2}$  chance of moving to the other. Our aim is to now show the eigenvalues of the normalized Laplacian,  $\bar{L}$ , are within  $\lambda$  of 1.

To do so, we will first show all eigenvalues of  $\bar{L}$  are  $\leq 1 + \lambda$ , and then demonstrate  $\bar{L}$  has second smallest eigenvalue  $1 - \lambda$ . Note that the second smallest eigenvalue is  $\min_{v \perp \mathbf{1}, \|v\|_2=1} v^\top \bar{L} v$ , and the largest eigenvalue is  $w^\top \bar{L} w$ , where  $w$  is the corresponding normalized eigenvector of this largest eigenvalue. Since  $\mathbf{1} \perp w$ , it suffices to show that for  $v \in \mathbb{R}^{2m}$  with  $\|v\|_2 = 1$  and  $v \perp \mathbf{1}$ , we have  $1 - \lambda \leq v^\top \bar{L} v \leq 1 + \lambda$ . WLOG assume the first  $m$  rows/columns are the vertices in  $C_0$  and the latter  $m$  are the vertices in  $C_1$ . Let  $t = \frac{1-\lambda}{1+\lambda}m$  and let  $v = (a_1, \dots, a_m, b_1, \dots, b_m)$ . Recall from the quadratic form version of the Laplacian and by construction of  $G$  that

$$(m+t)v^\top \bar{L} v = \sum_{1 \leq i < j \leq m} (a_i - a_j)^2 + \sum_{1 \leq i < j \leq m} (b_i - b_j)^2 + \sum_{\substack{1 \leq i \leq m \\ 0 \leq j < t}} (a_i - b_{i+j})^2$$

where indices are taken modulo  $m$ . Since  $v \perp \mathbf{1}$ , we have

$$\begin{aligned} 0 &= \left( \sum_{i=1}^m a_i + \sum_{i=1}^m b_i \right)^2 \\ &= 2m \left( \sum_{i=1}^m a_i^2 + \sum_{i=1}^m b_i^2 \right) - \sum_{1 \leq i < j \leq m} (a_i - a_j)^2 \\ &\quad - \sum_{1 \leq i < j \leq m} (b_i - b_j)^2 - \sum_{\substack{1 \leq i \leq m \\ 0 \leq j < m}} (a_i - b_{i+j})^2 \\ &= 2m\|v\|_2^2 - (m+t)v^\top \bar{L} v - \sum_{\substack{1 \leq i \leq m \\ t \leq j < m}} (a_i - b_{i+j})^2 \\ v^\top \bar{L} v &= \frac{2m}{m+t} - \frac{1}{m+t} \sum_{\substack{1 \leq i \leq m \\ t \leq j < m}} (a_i - b_{i+j})^2 \end{aligned} \tag{18}$$

For one side, we can trivially upper bound (18)

$$v^\top \bar{L} v \leq \frac{2m}{m+t} = 1 + \lambda.$$

For the lower bound of (18), we can use Cauchy-Schwarz to get

$$\begin{aligned} v^\top \bar{L} v &\geq \frac{2m}{m+t} - \frac{2}{m+t} \sum_{\substack{1 \leq i \leq m \\ t \leq j < m}} (a_i^2 + b_{i+j}^2) \\ &= \frac{2m}{m+t} - \frac{2m-2t}{m+t} \|v\|_2^2 \\ &= \frac{2m}{m+t} \\ &= 1 - \lambda. \end{aligned}$$

Hence we can conclude the nonzero eigenvalues of  $\bar{L}$  are within  $\lambda$  of 1. Thus,  $G$  is indeed a  $\lambda$ -expander that models a  $S(n, \lambda)$  sticky walk.  $\blacktriangleleft$

Note that this construction gives a family of  $\lambda$ -expanders: one for each  $m$  where  $\left(\frac{1-\lambda}{1+\lambda}\right)m$  is an integer. In order to extend the above theorem to degree-bounded expanders, we believe replacing the cliques  $C_0$  and  $C_1$  with degree  $d$  expanders, and adding a random bi-regular bipartite graph with degree  $\left(\frac{1-\lambda}{1+\lambda}\right)d$  between them will suffice, but we have not verified this.

---

## References

- 1 Louay Bazzi. Entropy of weight distributions of small-bias spaces and pseudobinomiality. *Chic. J. Theor. Comput. Sci.*, 2015, 2015. URL: <http://cjtcs.cs.uchicago.edu/articles/2015/2/contents.html>.
- 2 Gil Cohen, Noam Peri, and Amnon Ta-Shma. Expander random walks: A fourier-analytic approach. *Electron. Colloquium Comput. Complex.*, 27:163, 2020. URL: <https://eccc.weizmann.ac.il/report/2020/163>.
- 3 T.M. Cover and J.A. Thomas. *Elements of Information Theory*. John Wiley and Sons, 2006.
- 4 David Gillman. A chernoff bound for random walks on expander graphs. *SIAM J. Comput.*, 27(4):1203–1220, 1998. doi:10.1137/S0097539794268765.
- 5 Venkatesan Guruswami and Vinayak Kumar. Pseudobinomiality of the sticky random walk. *Electron. Colloquium Comput. Complex.*, 27:151, 2020. URL: <https://eccc.weizmann.ac.il/report/2020/151>.
- 6 A. Gut. *Probability: A Graduate Course*. Springer Texts in Statistics. Springer New York, 2006. URL: [https://books.google.com/books?id=4Mqk\\_udhw1kC](https://books.google.com/books?id=4Mqk_udhw1kC).
- 7 Nabil Kahale. Large deviation bounds for Markov chains. *Comb. Probab. Comput.*, 6(4):465–474, 1997. URL: <http://journals.cambridge.org/action/displayAbstract?aid=46567>.
- 8 Athanasios Papoulis and S. Unnikrishna Pillai. *Probability, Random Variables, and Stochastic Processes*. McGraw Hill, Boston, fourth edition, 2002. URL: [http://www.worldcat.org/search?qt=worldcat\\_org\\_all&q=0071226613](http://www.worldcat.org/search?qt=worldcat_org_all&q=0071226613).
- 9 Anup Rao. An exposition of Bourgain’s 2-source extractor. *Electronic Colloquium on Computational Complexity (ECCC)*, 14(034), 2007. URL: <http://eccc.hpi-web.de/eccc-reports/2007/TR07-034/index.html>.
- 10 Shravas Rao and Oded Regev. A sharp tail bound for the expander random sampler. *CoRR*, abs/1703.10205, 2017. arXiv:1703.10205.
- 11 Alex Samorodnitsky. On the optimum of Delsarte’s linear program. *J. Comb. Theory, Ser. A*, 96(2):261–287, 2001. doi:10.1006/jcta.2001.3176.

- 12 Amnon Ta-Shma. Explicit, almost optimal, epsilon-balanced codes. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 238–251. ACM, 2017. doi:10.1145/3055399.3055408.
- 13 Salil Vadhan. Personal communication, 2018.

## A

 Deferred Proofs

▷ **Claim 14.** For  $1 \leq k \leq n/2$ , we can bound  $\frac{\binom{n}{k}^2}{\binom{n}{2k}} \leq \frac{e^3 \sqrt{2}}{4\pi^2} \cdot 4^{2k}$ .

**Proof.** We can use Stirling's approximation  $\sqrt{2\pi} \left(\frac{n}{e}\right)^n \sqrt{n} \leq n! \leq e \left(\frac{n}{e}\right)^n \sqrt{n}$  to bound

$$\begin{aligned}
 \frac{\binom{n}{k}^2}{\binom{n}{2k}} &= \frac{n!(2k)!(n-2k)!}{k!^2(n-k)!^2} \\
 &\leq \frac{e^3}{4\pi^2} \cdot \frac{n^n (2k)^{2k} (n-2k)^{n-2k}}{k^{2k} (n-k)^{2n-2k}} \sqrt{\frac{2n(n-2k)}{k(n-k)^2}} \\
 &\leq \frac{e^3}{4\pi^2} \cdot 2^{2k} \left(\frac{n}{n-k}\right)^n \left(\frac{n-2k}{n-k}\right)^{n-2k} \sqrt{\frac{2}{k}} \\
 &\leq \frac{e^3 \sqrt{2}}{4\pi^2} \left(\frac{2n}{n-k}\right)^{2k} \\
 &\leq \frac{e^3 \sqrt{2}}{4\pi^2} \cdot 4^{2k}
 \end{aligned}
 \quad \triangleleft$$

► **Lemma 17.** As  $n \rightarrow \infty$ ,  $\mathbf{Y} \sqrt{\frac{1-\lambda}{(1+\lambda)n}} \rightarrow \mathbf{N}$  in distribution.

**Proof.** The idea is to use the method of moments (see Theorem 8.6 of [6]), which states the lemma can be deduced if all moments of  $\mathbf{Y}$  approach the moments of  $\mathbf{N}$  as  $n \rightarrow \infty$ . The odd moments of the Gaussian distribution is zero, and the  $2k$ 'th moment is  $(2k-1)!! := \frac{(2k)!}{2^k k!}$  for all  $k \geq 1$  (see [8]). Clearly  $\mathbb{E} \left[ \left( \mathbf{Y} \sqrt{\frac{1-\lambda}{(1+\lambda)n}} \right)^{2k+1} \right] = 0$  by the symmetry of  $\mathbf{Y}$ .

It now remains to show that  $\mathbb{E} \left[ \left( \mathbf{Y} \sqrt{\frac{1-\lambda}{(1+\lambda)n}} \right)^{2k} \right] \rightarrow (2k-1)!!$  as  $n \rightarrow \infty$ . Let  $P_k$  denote the set of unordered partitions of  $k$  into positive parts, where we express each partition as a multiset of positive integers  $A = [a_1, \dots, a_m]$  where  $\sum_{i=1}^m a_i = k$ . We define  $g([a_1, \dots, a_m]) = \frac{(\sum_{i=1}^m a_i)!}{\prod_{i=1}^m a_i!}$ . By expanding  $(\sum_{i=1}^n \mathbf{Y}_i)^{2k}$ , collecting terms with the same multiset of degrees, and taking exponents modulo 2,

$$\begin{aligned}
 \mathbb{E}[\mathbf{Y}^{2k}] &= \mathbb{E} \left[ \left( \sum_{i=1}^n \mathbf{Y}_i \right)^{2k} \right] = \sum_{P \in P_{2k}} g(P) \mathbb{E} \left[ \sum_{T \in \binom{[n]}{P}} \prod_{i=1}^{|P|} \mathbf{Y}_{t_i}^{p_i} \right] \\
 &= \sum_{P \in P_{2k}} g(P) \mathbb{E} \left[ \sum_{T \in \binom{[n]}{P}} \prod_{i: 2 \nmid p_i} \mathbf{Y}_{t_i} \right]
 \end{aligned}
 \quad (19)$$

where  $P = [p_1, \dots, p_{|P|}]$  and  $T = \{t_1, \dots, t_{|P|}\}$ . Now let  $P_o$  and  $P_e$  be the multiset containing all odd and even elements of  $P$  (with multiplicity), respectively. Define  $h([a_1, \dots, a_m])$  to

## 48:18 Pseudobinomiality of the Sticky Random Walk

be the number of ways to permute  $(a_1, \dots, a_n)$  (e.g.  $g([1, 2, 2, 6]) = 12$  since there are 12 ways of permuting  $(1, 2, 2, 6)$ ,  $g([2, 2, 2]) = 1$ , and  $g([1, 4, 5, 6]) = 24$ ). By Equation 5 and Lemma 12, we have

$$\mathbb{E} \left[ \sum_{T \in \binom{[n]}{|P_o|}} \prod_{i \in T} \mathbf{Y}_i \right] = \mathbb{E}[K_{|P_o|}(|\mathbf{s}|)] = \sum_{m=|P_o|/2}^{n-|P_o|/2} \binom{m-1}{|P_o|/2-1} \binom{n-m}{|P_o|/2} \lambda^m.$$

With this, we can rewrite

$$\begin{aligned} \mathbb{E} \left[ \sum_{T \in \binom{[n]}{|P|}} \prod_{i \in T} \mathbf{Y}_{t_i} \right] &= h(P_e) \binom{n-|P_o|}{|P_e|} \mathbb{E} \left[ \sum_{T \in \binom{[n]}{|P_o|}} \prod_{i \in T} \mathbf{Y}_i \right] \\ &= h(P_e) \binom{n-|P_o|}{|P_e|} \sum_{m=|P_o|/2}^{n-|P_o|/2} \binom{m-1}{|P_o|/2-1} \binom{n-m}{|P_o|/2} \lambda^m \\ &\sim h(P_e) \frac{n^{|P_e|}}{(|P_e|)!} \sum_{m=|P_o|/2}^{n-|P_o|/2} \binom{m-1}{|P_o|/2-1} \frac{n^{|P_o|/2}}{(|P_o|/2)!} \lambda^m \\ &= n^{|P|-|P_o|/2} \cdot \frac{h(P_e)}{(|P_o|/2)! (|P_e|)!} \sum_{m=|P_o|/2}^{n-|P_o|/2} \binom{m-1}{|P_o|/2-1} \lambda^m \end{aligned} \quad (20)$$

and so by combining (19) and (20), we have

$$\mathbb{E}[Y^{2k}] \sim \sum_{P \in P_{2k}} n^{|P|-|P_o|/2} \cdot \frac{g(P)h(P_e)}{(|P_o|/2)! (|P_e|)!} \sum_{m=|P_o|/2}^{n-|P_o|/2} \binom{m-1}{|P_o|/2-1} \lambda^m. \quad (21)$$

We just have to look at the leading term of this sum, which is when  $|P| - |P_o|/2$  is maximized (since  $|P_{2k}|$ ,  $g(P)$ , and  $h(P_e)$  don't depend on  $n$ ). Note

$$|P| - \frac{|P_o|}{2} = (|P_o| + |P_e|) - \frac{|P_o|}{2} = \frac{|P_o| + 2|P_e|}{2} \leq \frac{\sum_{p \in P_o} p + \sum_{p \in P_e} p}{2} = k.$$

Hence the leading terms are when  $|P| - |P_o|/2 = k$ , and these terms correspond to when all elements of  $P$  are 1 or 2. In particular, the leading terms correspond to the multisets  $P$  with  $2r$  1's and  $k-r$  2's for  $0 \leq r \leq k$ . In these cases, we can calculate  $|P| = k+r$ ,  $|P_o| = 2r$ ,  $h(P_e) = 1$  and  $g(P) = \frac{(2k)!}{2^{k-r}}$ . Hence from (21) we get

$$\begin{aligned} \mathbb{E}[Y^{2k}] &\sim \sum_{r=0}^k \frac{(2k)!}{2^{k-r} r! (k-r)!} \left( \sum_{m=r}^{n-r} \binom{m-1}{r-1} \lambda^m \right) n^k \\ &= (2k-1)!! \sum_{r=0}^k \binom{k}{r} 2^r \left( \sum_{m=r}^{n-r} \binom{m-1}{r-1} \lambda^m \right) n^k. \end{aligned}$$

Consequently, we have the  $2k$ 'th moment of  $\mathbf{Y} \sqrt{\frac{1-\lambda}{(1+\lambda)n}}$  is

$$\sim (2k-1)!! \left( \frac{1-\lambda}{1+\lambda} \right)^k \sum_{r=0}^k \binom{k}{r} 2^r \left( \sum_{m=r}^{n-r} \binom{m-1}{r-1} \lambda^m \right)$$

Taking  $n \rightarrow \infty$  and evaluating the series with the well-known generating function identity  $(\frac{x}{1-x})^r = \sum_{m \geq r} \binom{m-1}{r-1} x^m$ , we get that the  $2k$ 'th moment approaches

$$\begin{aligned}
 (2k-1)!! \left( \frac{1-\lambda}{1+\lambda} \right)^k \sum_{r=0}^k \binom{k}{r} 2^r \left( \sum_{m=r}^{\infty} \binom{m-1}{r-1} \lambda^m \right) \\
 &= (2k-1)!! \left( \frac{1-\lambda}{1+\lambda} \right)^k \sum_{r=0}^k \binom{k}{r} \left( \frac{2\lambda}{1-\lambda} \right)^r \\
 &= (2k-1)!! \left( \frac{1-\lambda}{1+\lambda} \right)^k \left( 1 + \frac{2\lambda}{1-\lambda} \right)^k \\
 &= (2k-1)!! .
 \end{aligned}$$

Hence by the method of moments  $\mathbf{Y}_{\sqrt{\frac{1-\lambda}{(1+\lambda)n}}} \rightarrow \mathbf{N}$  in distribution as  $n \rightarrow \infty$ . ◀





# Robust Quantum Entanglement at (Nearly) Room Temperature

Lior Eldar

Yifat, Israel

eldar.lior@gmail.com

---

## Abstract

We formulate an average-case analog of the NLTS conjecture of Freedman and Hastings (QIC 2014) by asking whether there exist topologically ordered systems with corresponding local Hamiltonians for which the thermal Gibbs state for constant temperature cannot even be approximated by shallow quantum circuits. We then prove this conjecture for nearly optimal parameters: we construct a quantum error correcting code whose corresponding (log) local Hamiltonian has the following property: for nearly constant temperature (temperature decays as  $1/\log^2 \log(n)$ ) the thermal Gibbs state of that Hamiltonian cannot be approximated by any circuit of depth less than  $\log(n)$ , and it is highly entangled in a well-defined way. This implies that appropriately chosen local Hamiltonians can give rise to ground-state long-range entanglement which can survive without active error correction at temperatures which are nearly independent of the system size: thereby improving exponentially upon previously known bounds.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Quantum complexity theory

**Keywords and phrases** Quantum error-correcting codes, Quantum Entanglement, Quantum Locally-Testable Codes, Local Hamiltonians, quantum PCP, NLTS

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.49

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/1911.04461v3>.

**Acknowledgements** The author thanks Dorit Aharonov, Simon Apers, Aram Harrow, Matthew Hastings and Anthony Leverrier for their useful comments and suggestions. He also thanks anonymous reviewers for their helpful comments and suggestions.

## 1 General

In order to perform universal quantum computation, one should at the very least be able to store quantum states for long periods of time. While the Fault Tolerance theorem [1] makes this possible using active error correction, in parallel, and in part due to the limitations of the FT theorem (see e.g. [4]) a huge research effort was devoted to finding quantum systems that can retain quantum information passively - namely a self-correcting quantum memory.

Self-correcting quantum memories are often referred to as topologically-ordered systems (or TQO) which is a phase of matter that exhibits long-range entanglement at 0 temperature. Since 0 temperature states are essentially theoretical objects that one does not expect to encounter in the lab, the race was on to find TQO systems whose long-range entanglement can survive at very high temperatures - ideally at a constant temperature  $T > 0$  that is independent of the system size.

In recent years there has been progress in ruling out such robustness for low-dimensional systems like the 2-D and 3-D Toric Code, but there has been an indication that perhaps in 4 dimensions and above, robustness is more likely (see Section 2.1.2 for a summary of these results). Intriguingly, it seems that quantum mechanics does not fundamentally limit the ability to store quantum states for long times, at least for high-dimensional systems. Despite



© Lior Eldar;

licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 49; pp. 49:1–49:20

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

that there remains today a large gap between our physical intuition and our ability to provide formal proofs on the existence of robust systems. Hence the problem of establishing the existence of robust TQO systems, even for high dimensions is wide open.

## 2 Topological Quantum Order (TQO)

TQO is a phase of matter (i.e. in addition to the traditional gas, liquid, and solid states) defined as the zero eigenspace of a local Hamiltonian which is “robust” in the sense that there can be no transition between orthogonal zero eigenstate without a phase transition (See survey of TQO in [19]). Formally one says that a system is  $\varepsilon$ -TQO if any sufficiently local observable  $O$  is unable to discern orthogonal states of the groundspace - i.e. there exists some  $z \neq 0$  such that

$$\|POP - zP\| \leq \varepsilon$$

where  $P$  is the projection onto the groundspace.

In the language of quantum computing, TQO is mostly synonymous with quantum error correcting codes, and specifically topological quantum codes - these are  $\varepsilon$ -TQO systems with  $\varepsilon = 0$ . Under error-correction terminology the TQO property of robustness of ground-state degeneracy is the quantum error-correcting minimal distance: i.e. the system can retain its logical encoded state in the presence of sufficiently small errors.

Thus, TQO systems have the promise that at zero temperature, their entanglement can passively sustain itself (i.e. without active error-correction) as a form of self-correcting quantum memory. It is this stability that brought forth the immensely influential paradigm of the topological quantum computer by Kitaev [14, 8], and even initiated large-scale engineering efforts in trying to build such a set-up [9].

However, since a physicist attempting to prepare a TQO state in a lab can only expect to encounter a Gibbs state (i.e. a thermal state) of the Hamiltonian governing the TQO for some low temperature, then for TQO to serve as a self-correcting quantum memory a necessary property for such a system is that it retains its long-range entanglement at some non-zero temperature  $T > 0$  that is independent of the size of the system.

A natural treatment of robustness of TQO systems can be made using quantum circuit lower bounds, a form of analysis initially considered in [11]. Under TQO terminology, topologically ordered states *cannot* be generated from a tensor product state using a shallow circuit, whereas a state is said to be “trivial” if it *can* be generated from product states by shallow circuits - namely it is nearly equivalent to a product state in its lack of quantum entanglement. With this terminology in mind we consider the following conjecture:

► **Conjecture 1** (Robust Circuit Depth for Topologically Ordered Systems). *There exists a number  $\beta > 0$  and a family of topologically-ordered systems (local Hamiltonians)  $\{H_n\}_n$  on  $n$  qubits such that for all  $\gamma \geq \beta$  we have: Any quantum circuit  $U$  that approximates the thermal Gibbs state  $e^{-\gamma H}$  to vanishing trace distance has depth  $\Omega(\log(n))$ .*

In words: the conjecture posits the existence of a TQO system (say, a quantum error-correcting code) for which one can show a circuit lower bound for the thermal state for all  $T$  from 0 (i.e. the ground-state) up to some constant temperature. Such a system exhibits “robustness” in the sense that the circuit lower bound for approximating its Gibbs state does not collapse when temperature is increased, with the logarithmic bound potentially allowing the coupling of *every* pair of qubits in the system, using a locally-defined quantum circuit.

Similar variants of this conjecture have been studied in physics literature: for example, in [20] Yoshida provides a negative solution to a similar conjecture for codes embeddable in 2 or 3 dimensional lattices. On the other hand, in [11, 13] the authors provided an indication to the affirmative of this conjecture by considering the 4-dimensional Toric Code: for example, Hastings [11] assumes the existence of certain error operators from which he derived a related property. In [3] the authors use an approximation of thermal systems called the weak Markovian limit, and conclude that certain topological measurements are preserved at constant temperatures for exponentially long time in the system size (exponential coherence times). However, to the best of our knowledge there is no formal proof of conjecture 1.

Here, we make progress towards affirming this conjecture formally by proving conjecture 1 for nearly-optimal parameters:

► **Theorem (sketch).** (Robust TQO at nearly Room Temperature). *There exists a log-local family of quantum error-correcting codes  $\{\mathcal{C}_n\}_n$  with polynomial minimal distance  $n^{\Omega(1)}$  (in particular, a topologically-ordered system) and a corresponding family of commuting local Hamiltonians  $\{H_n\}_n$  such that for any  $\beta_n = O(\log^2 \log(n))$  the following holds: any quantum circuit  $V$  that acts on a  $\geq n$  qubits and approximates the thermal state at temperature at most  $T_n = 1/(\kappa\beta_n)$  on a set of qubits  $S$ ,  $|S| = n$ :*

$$\left\| \frac{1}{Z} e^{-\beta_n H_n} - \text{tr}_{-S}(V|0^{\otimes a}\rangle\langle 0^{\otimes a}|V^\dagger) \right\|_1 = o(1), \quad Z = \text{tr}(e^{-\beta_n H_n})$$

satisfies a circuit lower bound:

$$d(V) = \Omega(\ln(n)).$$

Our proof will actually show a stronger statement: namely that the thermal Gibbs state of these codes, for sufficiently low, yet nearly constant temperature, can be decoded using a shallow circuit to a bona-fide quantum code state. This implies that the Gibbs state retains topological order (up to a shallow decoder) at very high temperatures: if we initialize our system in some ground-state of the TQO, and allow it to thermalize, we can later recover that code-state with little extra cost (see Section 2.1.7 discussing the possible implementation error of such a set-up). In particular, it implies that appropriately chosen local Hamiltonians can give rise to ground-state multi-partite entanglement which can survive without active error correction at nearly-constant temperatures.

## 2.1 Some Perspective

### 2.1.1 The Thermal Gibbs State

This study explores quantum circuit lower bounds on arguably the most natural of physical states - namely the thermal Gibbs state  $e^{-\beta H}$  - which can be formed by coupling a ground-state of a physical system  $H$  to a “heat bath” at temperature  $1/\beta$  - meaning it is allowed to interact indefinitely with an environment to which we have no access to. (see definition 3)

### 2.1.2 The Regimes of “Inverse-Temperature” $\beta$

We consider here a summary of prior art: the temperature at which one can establish an  $\Omega(\log(n))$  (i.e. “global”) circuit lower bound for the Gibbs state of a Hamiltonian:

Hamiltonian	Temperature	Result	Comments
QECC with large distance	0	Folklore	
2 or 3-D systems	$O(1/\text{poly}(n))$ <sup>1</sup>	[20]	No-go theorem
Projective Code	$\Omega(1/\text{polylog}(n))$	This work	By definition, Without amplification
Amplified Projective Code	$\Omega(1/\text{polyloglog}(n))$	This work	With amplification
4-D Toric Code	$\Omega(1)$	[11]	Heuristic argument.

Our main theorem establishes the existence of log-local Hamiltonians for which the thermal state  $e^{-\beta H}$  for  $\beta = (\log\log(n))^2$  requires a logarithmic circuit depth. Therefore it improves exponentially on previous work in terms of the provable highest temperature as a function of system size  $n$  at which circuit lower bounds can be maintained.

Notably, observe that the rate of errors experienced by quantum states from this ensemble scales like  $n/\text{polylog}(n)$  - i.e. a nearly linear fraction. Such error rates result in error patterns whose weight is much larger than the minimal error-correcting distance of the quantum code, and hence it is not immediately clear, at least from an information-theoretic perspective, whether these states - that formally cannot protect quantum information - can be assigned circuit lower bounds.

### 2.1.3 Is it Entangled?

Any quantum system satisfying Conjecture 1 has a highly entangled ground-space, because for  $T = 0$  a Gibbs state can be any (possibly pure) ground-state of a topologically ordered system. That said, for  $T > 0$  the circuit lower bound is applied to mixed states: assigning a quantum circuit lower bound for the task of approximating a quantum mixed state (as opposed to a pure state) does not necessarily indicate the existence of quantum correlations but rather the presence of long-range correlations, which may or may not be quantum. In fact, by using the argument of Lovett and Viola [17] one can show a  $\Omega(\log(n))$  circuit lower bound for the Gibbs state at constant temperature  $T = \Omega(1)$  of any good classical locally testable code, though, this bound breaks for  $T = 0$  since any classical string codeword, which is unentangled, is a valid Gibbs state.

However, as noted above even this somewhat weaker notion of a quantum system with a highly-entangled ground-space that retains a quantum circuit lower bound at very high temperature isn't known to exist (at least formally), and is related to major open questions in quantum complexity theory. See in this context the NLTS conjecture discussed in the section 2.1.6.

### 2.1.4 Quantum Circuit Lower Bounds

This work adds to the set of available tools for showing quantum circuit lower bounds - by combining quantum locally-testable codes, an analysis of the thermal state as a truncated Markov chain, and a local decoder that relies on these two properties to decode a thermal state to a bona-fide quantum code-state which can be assigned a circuit lower bound by information-theoretic arguments. Previous works have either used quantum locally testable codes [5] to argue direct circuit lower bounds, or local decoders [11, 16] but as far as we know these strategies were never used in conjunction. We outline our strategy in more detail in Section 2.3

### 2.1.5 Energy versus error

An important distinction that one needs to make early on is that having a quantum state with low-energy *does not* necessarily imply it is generated by applying few errors to a ground-state. This is only true if the Hamiltonian governing the quantum state is a so-called qLTC [2]. qLTC's are quantum analogs of locally-testable codes (and see Definition 7).

Like their classical counterparts qLTC's are (local) Hamiltonians for which large errors necessarily result in a large number of violations from a set of local check terms. To give an example - consider Kitaev's 2-dimensional Toric Code [14] at very low-temperature, say  $T = O(1/\sqrt{n})$ . At that temperature the probability of an error of weight  $\sqrt{n}$ , at least one which is composed of strings of weight  $\sqrt{n}$  is proportional to the probability of observing an error of constant energy, i.e.:  $e^{-\beta O(1)}$  i.e. comparable to the probability of a single error. So, unless additional structure of the problem is used, for all we know the number of errors could be  $\Omega(n)$ . The reason for the above is that the Toric Code is known to have very poor soundness as a locally testable code: in fact one can have an error of size  $\sqrt{n}$  with only two violations.

### 2.1.6 The relation to NLTS

Conjecture 1 above is a mixed-state analog (albeit with a slightly more stringent requirement on the circuit depth) of the NLTS conjecture due to Freedman and Hastings [7] - which posits the existence of local Hamiltonians for which *any* low-energy state can only be generated by circuits of diverging depth. As far as we know neither conjecture is stronger than the other.

Arguably, the only known strategy to establishing the NLTS conjecture, outlined in [5], is to show a construction of quantum locally-testable codes (qLTC's) with constant soundness and linear minimal quantum error correcting distance. However, such a statement by itself requires the construction of quantum LDPC codes with distance growing linearly in the number of qubits - a conjecture now open for nearly 30 years. Thus our inability to make progress on qLDPC is a significant barrier to any progress on the NLTS conjecture.

In this work, we show that by considering a mixed-state (or average-case) analog of NLTS (while still placing a more stringent requirement on the circuit depth) one can break away from this strategy using the tools we already have today - namely qLTC's with  $1/\text{polylog}$  soundness and code distance which is sub-linear in  $n$ , in this case  $\sqrt{n}$ , and achieve a construction with nearly optimal parameters. Nevertheless, it could be the case that the construction provided here is in fact NLTS - meaning there are no trivial states of the code Hamiltonian for sufficiently small constant temperature.

### 2.1.7 Implementation Error

Above, we mentioned the ability of the proposed system to allow thermalization of an initialized state, and still be able to recover that state using a local decoder. Arguably, one can argue against such a statement that one also needs to account for the implementation error of the Hamiltonian governing the TQO state, as well as the decoding Hamiltonian. There exist analogous claims against active fault-tolerance in the form of implementation error of the error-correcting unitary circuits.

However, we conjecture that the system we construct, insofar as the check terms of the Hamiltonians are concerned  $\{C_i\}$ , is in fact robust against implementation error by virtue of local testability. Recall that a locally testable code (see Definition 7) satisfies the following operator inequality:  $\frac{1}{m} \sum_{i=1}^m C_i \succeq \frac{s}{n} D_C$ . where  $D_C$  is an operator that relates a state to its distance from the code-space.

One can check that if the LHS above suffers from an additive error quantified by a Hermitian error operator  $\mathcal{E}$ ,  $\frac{1}{m} \sum_{i=1}^m C_i + \mathcal{E}$  then using standard results about stability of Hermitian operators under Hermitian perturbation, the resulting code will still be, for sufficiently weak error  $\mathcal{E}$ , a locally testable code albeit with slightly worse parameters, and for a smaller range of distances from the codespace: it will not be able to faithfully test very small errors, but only large errors.

Still, this code will possess the key property that we use here to argue the main theorem: that for sufficiently low temperature (depending on the strength of the implementation error  $\|\mathcal{E}\|$ ) - the code reins in the error weight to small weights, and these errors are far apart to allow for local error correction. Notably, the actual shallow decoder used in the argument will also suffer from implementation noise, undoubtedly, but as a theoretic argument about entanglement, it is only important to account for implementation error of the Hamiltonian, and not the decoding circuit.

Hence the system proposed has apparently two advantages: not only it is able to sustain long-range entanglement for high temperatures as established in Theorem 31, one doesn't even need to implement it precisely to gain the first advantage. We leave for future research to quantify precisely the degree to which this system is robust against implementation error.

## 2.2 Some Open Questions

We end this section with several questions for further research. First, it is desirable to improve (reduce) the value of  $\beta$  and improve (reduce) the locality of checks (currently they are log-local). We note that a limiting factor to decreasing  $\beta$  is the maximal size set for which one can show near-optimal expansion, for qLTC's.

An interesting extension of this work is to extend it to actual quantum information - namely show the system can store an arbitrary quantum state for long periods of time: notably here we have only showed that one can recover the uniform distribution on code-states, but it is not immediately clear that it can preserve a single arbitrarily encoded code-state. In addition, it would be insightful to understand the actual coherence time of such a system as a self-correcting quantum memory - we conjecture that it is polynomial in  $n$ .

Finally, one could explore the possibility that the constructed code in fact satisfies the NLTS condition: namely that any low-energy state is highly entangled.

## 2.3 Overall Strategy

In figure 1 we outline the main steps of our argument. To recall, the main goal of this study is to demonstrate that the thermal state  $e^{-\beta H}/Z$  is hard to produce for sufficiently small  $\beta$ , and show the same for any ground-state of  $H$  - mixed or pure.

Our overall strategy is to demonstrate a *shallow* quantum circuit that allows to correct this thermal state to some code-state of a quantum code with large minimal distance. For a quantum code with large minimal error-correcting distance it is a folklore fact (made formal here) that any quantum state in that codespace is hard to approximate (the gray-shaded box in Figure 1), thereby satisfying the hardness-of-approximation requirement for groundstates. Furthermore, together with the existence of a shallow decoder, it implies a lower-bound on the circuit depth for  $e^{-\beta H}$  as the lower-bound on a circuit generating a quantum code-state, minus the depth of the decoder. Thus, working in the diagram of Figure 1 backwards we translate our overall theorem to demonstrating a shallow error-correcting circuit from a thermal state to a code-state with polynomial distance.

### 2.3.1 Translating Energy to Error

The strategy outlined above requires us to demonstrate a shallow decoder for thermal states of sufficiently low temperatures. Here we are faced with a severe obstacle: a thermal Gibbs state is defined in terms of energy, whereas the natural language for decoders is the language of “errors” (whether they are average-case or worst-case). Hence we need a scheme to argue about the error distribution of the Gibbs state.

To our aid come quantum locally testable codes (qLTCs) [2] (and see Definition 7). The main use of locally testable codes is to rein in the error weight of low-energy states. We use this property in conjunction with the well-known Metropolis-Hastings algorithm (or MH) on Hamiltonians corresponding to the check terms of qLTC’s. The MH algorithm is a standard tool in physics to simulate the thermal Gibbs state by a random walk where transition probabilities between quantum states are dictated by their relative energies (see Section 7).

Applying this tool to local Hamiltonians corresponding to qLTCs we show that the thermal Gibbs state  $e^{-\beta H}$ , for  $H$  corresponding to a qLTC, can in fact be approximated by a so-called “truncated” MH process. These arguments correspond to the top Vanilla-colored boxes in the diagram.

As a general note, as far as we know, no previous work using qLTC’s made such a translation from energy to error weight: in [5] the authors show that qLTC’s with linear distance are NLTS, but since such codes are not known to exist, they end up proving a somewhat weaker version called NLETS thus bypassing the energy-to-error translation. On the other hand, such a translation is probably the most natural way to proceed w.r.t. quantum codes: we do not know how to treat “energies” on quantum states, but if we can model the errors they experience we can leverage our vast knowledge of quantum *error-correction* to handle them. Hence, we believe that the use of the MH random process is of conceptual importance and will be useful elsewhere, since it allows for the first time, to bring the analysis from a point we want to argue about (“energies”) to a point where we have powerful analysis tools (“errors”).

### 2.3.2 Shallow Decoding from Local Expansion

To recap the flow of arguments: the arguments about the MH random process (Vanilla-colored boxes in Figure 1) allow us to argue that the thermal Gibbs state  $e^{-\beta H}$ , for a qLTC Hamiltonian  $H$ , can be simulated by sampling an error according to an MH random walk that is controlled by the inverse temperature  $\beta$  and the soundness of the qLTC. We would now like to leverage that property to demonstrate a shallow decoder for this state.

A key observation towards that end (corresponding to the bottom Vanilla-colored box) is that while the MH random process is not an i.i.d. process, it does in fact conform to a somewhat weaker characterization called “locally-stochastic” (or “locally-decaying”) [15, 6, 10], which are a main source of inspiration of this work. A noise is locally-stochastic if the probability of a large cluster of errors decays exponentially in its size (see Definition 23).

Concretely, using the truncated MH random process we conclude (orange box) that for sufficiently large  $\beta$  (low temperature) the errors experienced by the Gibbs state are locally-stochastic, and hence typically form only small clusters whose size is, say, at most  $\log(n)$ .

In effect, a stronger notion is true: we show that local-stochasticity of these random errors means that their clusters are also far away from each other - in the sense that even if we “blow up” each cluster by a factor of  $1/\alpha$  (for small  $\alpha > 0$ ) they are still at most the size above. This definition is called  $\alpha$ -subset and it too, is due to [15, 6].



In these works, error patterns that are locally-stochastic were shown to be amenable to correction by a local decoder, since intuitively, these errors can be “divided-and-conquered” locally. In this context, the notion of  $\alpha$ -subsets was used to handle the possibility that the decoder can introduce errors to qubits which weren’t initially erred, by arguing that even if such an event occurs it will not cause the initial clusters to aggregate together to form large, undecodable clusters.

Our choice for a shallow “local-decoder” is to use a straightforward quantum generalization of the Sipser-Spielman decoder (notably, a variant of this decoder was used in [15]). This decoder is desirable since it is able, under certain conditions to decode an error of weight  $w$  in time  $\log(w)$ , and do so locally. That would imply that for error patches of logarithmic size, the decoder would run in depth  $O(\log\log(n))$  - i.e. a very shallow decoder.

However, such a decoder comes attached with a very stringent condition: it requires the Tanner graph of the code to be a very strong bi-partite expander. That condition is too stringent for our purposes, since we also need the quantum code to be locally-testable, and it is not known how to make even classical local-testability co-exist with the code’s Tanner graph being a bi-partite expander.

In our study, we relax the stringent expansion condition, and require that only very small errors, i.e. those of logarithmic size which we’ve shown to be the typical error size for the Gibbs state - those errors are required to expand well (“small-set expansion”), while requiring nothing for linear-weight errors, which is the regime of interest of the standard Sipser-Spielman decoder. Hence, we are able to use a code whose Tanner graph is not a true expander. This while still being able to use the Sipser-Spielman approach to a fast parallel decoder by considering only small sets. These arguments are outlined in the pink-shaded boxes in the middle of the diagram.

## 2.4 The construction

To recap again, starting from the previous section, our goal is to find quantum a code, whose thermal state can be corrected quickly and in parallel. We’ve shown that if a code is qLTC then the Gibbs state can be essentially modeled as an error process that is locally-stochastic. Locally-stochastic errors can be decoded quickly, if the underlying topology is a good expander - at least for the typical error size. Hence, our interim goal is to find a quantum code  $\mathcal{C}$  that satisfies simultaneously three requirements:

1. It has a minimal quantum error-correcting distance that is some polynomial in the number of qubits  $n$ , say  $\sqrt{n}$  - to allow a circuit lower bound for proper code-states.
2. It is *locally-testable* - to allow translation from energies to errors in the truncated-MH modeling of the Gibbs state.
3. Expansion of the bi-partite Tanner graph corresponding to the checks of the code, for errors of small weight (or “small-set expansion”) - to allow for shallow decoding using the Sipser-Spielman algorithm.

### 2.4.1 The Choice of Quantum Code

In [12] Hastings found a way to make progress on the qLTC conjecture [2] by considering high-dimensional manifolds: he showed that by tessellating a high-dimensional sphere using a regular grid (or some other topology for improved rate) the resulting quantum code on  $n$  qubits has soundness  $1/\text{polylog}(n)$ . We make a crucial use of his approach here.

Recently, Leverrier et al. [16] have proposed the projective code, which is an arguably simpler variant of this high-dimensional construction whereby a length-3 complex chain of  $p$ -faces of the binary  $N$ -cube (modulo the all-ones vector), for  $p = \Omega(N)$  is used to derive

a quantum code on  $n$  qubits with distance scaling like  $n^c$ , for some constant  $c > 0$ . This code has improved soundness compared to the one in [12]. Our construction is based on the projective code on  $n$  qubits, using  $p$ -faces of the  $N$ -dimensional cube for  $p = N/2$ , where  $N = \Theta(\log(n))$ .

On one hand, by the minimal distance of the projective code one immediately gains a circuit lower bound of  $\Omega(\log(n))$  on the minimal depth circuit generating its ground-state (corresponding to the gray-shaded block in Figure 1. This satisfies the first requirement above. It is also a qLTC with reasonable  $(1/\log^2(n))$  soundness, (see navy-shaded block in Figure 1) thus satisfying the second requirement.

However, the last critical advantage that we gain by using this code, as opposed to say the original high-dimensional manifold of Hastings, is not its improved soundness parameter but rather the underlying structure of the high-dimensional cube: namely its property of small-set expansion that exists in addition to its non-negligible soundness. This property of small set expansion is the turnkey for allowing the application of a shallow decoder to combat the typical errors of a thermal state with large  $\beta$  parameter.

More specifically, we make crucial use of the structure of the  $N$ -cube to establish the third requirement - namely, show that small error sets expand significantly - i.e. have many unique incident constraints. This emanates from the fact that small subsets of  $p$ -faces of the  $n$ -cube for  $p = n/2$  have many adjacent  $p + 1$ -faces and many adjacent  $p - 1$ -faces.

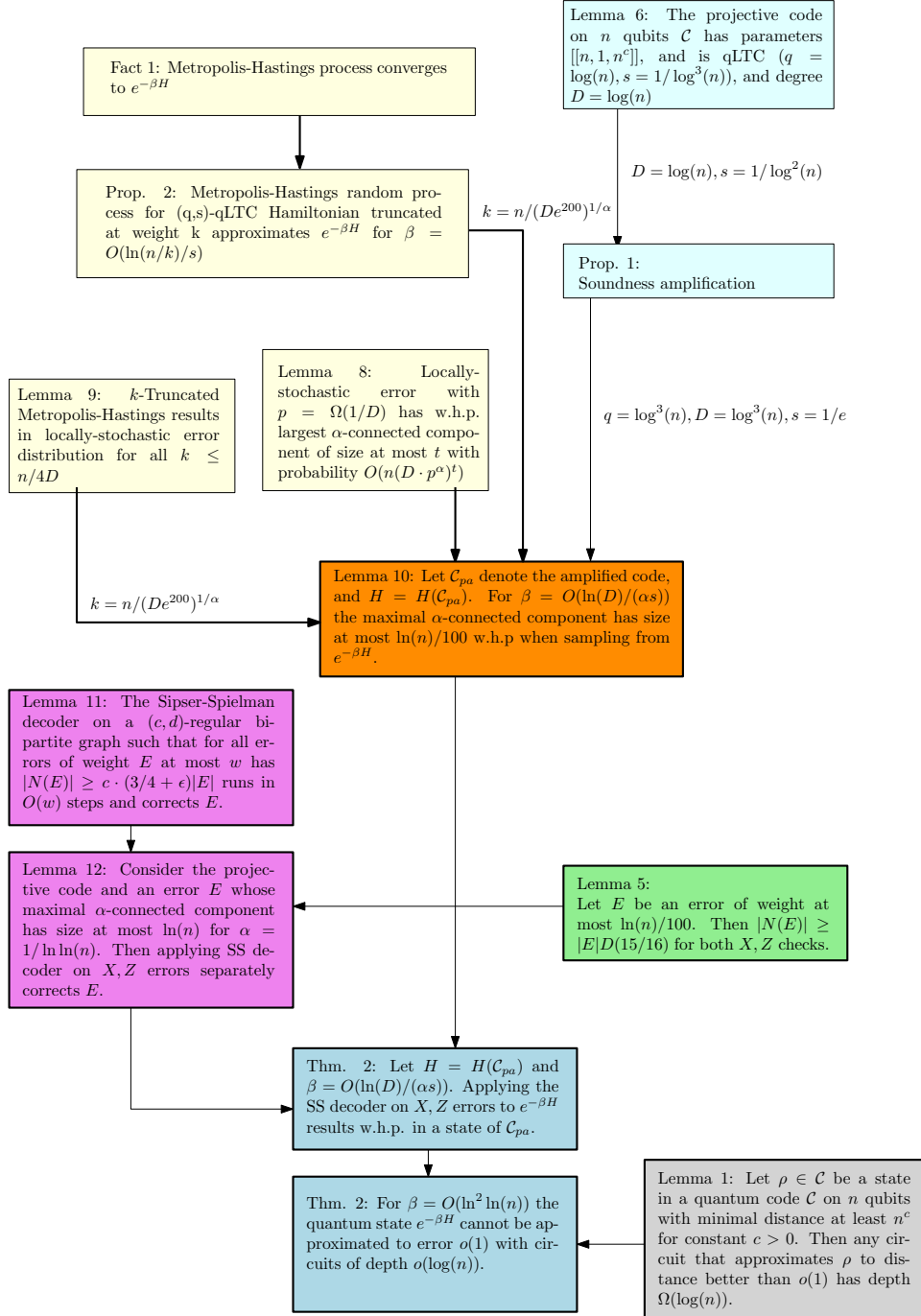
## 2.4.2 Efficient Soundness Amplification

The flow of arguments until this point results in lemma 27 which roughly states that a qLTC with soundness  $s$  and qubit degree  $D$  has the property that its thermal state for sufficiently large inverse temperature:  $\beta \geq \log(D)/s$ . has error patterns that form clusters of only logarithmic size. Such errors admit a shallow-depth parallel decoding scheme resulting in a circuit lower bound for approximating this thermal state.

Consider, for example the projective code of [16]. We have  $D = \log(n)$ ,  $s = 1/\log^2(n)$ . Using these parameters one would only be able to establish a circuit lower bound for  $\beta = \text{polylog}(n)$ . Hence, by the behavior of  $\beta$  as a function of  $D$  and  $s$  one sees it is desirable to trade-off increased degree for improved soundness so long as these two quantities are increased/decreased in a commensurate manner.

Using standard probabilistic analysis we show that it is sufficient to choose a random family of  $\Omega(n\log^2(n))$  subsets - each set comprised of  $1/s$  checks in order to achieve a qLTC with constant soundness and query size  $q/s$ . This corresponds to the second navy-shaded box in Figure 1.

This amplification procedure results in a somewhat peculiar situation that we'd like to point out: the thermal Gibbs state  $e^{-\beta H}$  is defined w.r.t. the Hamiltonian  $H = H(\mathcal{C}_{pa})$  where  $\mathcal{C}_{pa}$  is the result of the amplification of the projective code formed by choosing a sufficiently large random family of subsets of check terms of size  $1/s$  each. However, the decoding procedure, using the Sipser-Spielman algorithm uses the original checks of  $\mathcal{C}$  to locate and correct errors, and not the amplified ones: this is because we do not establish local expansion for the amplified checks, only for the original checks. Still, both sets of checks share the same code-space - namely the original projective code  $\mathcal{C}$ . Hence, the set of checks used for *testing* are not the same as the ones used for *correcting errors*.



■ **Figure 1** Flow of the main arguments.

### 3 Notation

A quantum CSS code on  $n$  qubits is a pair of codes  $\mathcal{C} = (\mathcal{C}_x, \mathcal{C}_z)$ , where  $\mathcal{C}_x, \mathcal{C}_z$  are subspaces of  $\mathbb{F}_2^n$ . For a thermal state  $\rho = (1/Z)e^{-\beta H}$ ,  $\beta$  signifies the “inverse temperature”  $\beta = 1/(\kappa T)$  where  $\kappa$  is the Boltzmann constant, and  $Z = \text{tr}(e^{-\beta H})$  is the partition function of this state. For a finite discrete set  $S$ ,  $|S|$  denotes the cardinality of the set. For  $\mathcal{E} \in \{0, 1\}^n$  the support of  $\mathcal{E}$ ,  $\text{supp}(\mathcal{E})$  is the set of non-zero positions of  $\mathcal{E}$ .  $|\mathcal{E}|$  is the Hamming weight of  $\mathcal{E}$ . For quantum density matrices  $A, B$ , the trace distance between  $A, B$  is denoted by  $\|A - B\|_1$ , and the quantum fidelity between these states is denoted by  $\mathcal{F}(A, B)$ . A density matrix  $\rho$  of rank  $r$  is said to be a *uniform mixture* if it can be written as  $\rho = \frac{1}{r} \sum_{i=1}^r |u_i\rangle\langle u_i|$  where  $\{|u_i\rangle\}$  are an orthonormal set of vectors.

We say that a quantum circuit  $U$  on a set  $T$  of  $N$  qubits approximates a quantum state  $\rho$  on a set  $S \subseteq T$  of  $n \leq N$  qubits, to error  $\delta$  if  $\|\text{tr}_{T-S}(U|0^{\otimes n}\rangle\langle 0^{\otimes n}|U^\dagger) - \rho\|_1 \leq \delta$

In this work, we will consider random error models  $\mathcal{E}$  supported on the  $n$ -th fold tensor product Pauli group  $\mathcal{P}^n$ , where  $\mathcal{P} = \{X, Z, Y, I\}$ . For an error  $\mathcal{E} = \mathcal{E}_1 \otimes \dots \otimes \mathcal{E}_n$ ,  $\mathcal{E}_i \in \mathcal{P}$  we denote by  $|\mathcal{E}|$  the Hamming weight of  $\mathcal{E}$  - namely the number of terms  $\mathcal{E}_i$  that are not equal to  $I$ . Often we will use  $|\mathcal{E}|$  to denote the minimal weight of  $\mathcal{E}$  modulo a stabilizer subgroup of  $\mathcal{P}^n$ . For a stabilizer code  $\mathcal{C}$  with local check terms  $\{C_i\}_{i=1}^m$ ,  $C_i \in \mathcal{P}^n$ , the Hamiltonian  $H = H(\mathcal{C})$  is the local Hamiltonian  $\sum_{i=1}^m (I - C_i)/2$  - i.e. its ground-space is the intersection of the 1-eigenspaces of all check terms  $C_i$ .

The  $N$ -cube is the binary cube in  $N$  dimensions. We will use capital  $N$  to denote the dimension of the cube. The projective cube results in a code of  $n$  qubits. When considering  $n$  in the context of the projective cube we will use lower-case  $n$  to denote the number of qubits, i.e.  $n = 2^N$ .

The letter  $a$  will be used to denote an initial set of qubits  $a \geq n$  that also include any ancillary qubits used to generate the state of  $n$  qubits, i.e. to generate a mixed state on  $n$  qubits one applies a unitary transformation on the state  $|0^{\otimes a}\rangle\langle 0^{\otimes a}|$ , traces out  $a - n$  qubits.

Let  $G = (V, E)$  be a graph. For a set  $S \subseteq V$  the set  $\Gamma(S) \subseteq V$  is the set of all vertices that neighbor  $S$  in  $G$ . The degree of a vertex  $v \in V$  is the number of edges incident on that vertex. The degree  $D$  of a graph is the maximal degree of any vertex  $v \in V$ . A graph is  $D$ -regular if the degrees of all vertices are equal. We will use  $a \propto b$  to signify that  $a = c \cdot b$  for some  $c$  that does not depend on  $b$ . Throughout the paper we will use the binary logarithm  $\log(n)$  in the context of quantum circuit lower bounds, and the natural logarithm  $\ln(n)$  in the context of errors occurring on a thermal state  $e^{-\beta H}$ .

### 4 Preliminaries I: Thermal Gibbs State of a Local Hamiltonian

When considering the thermal Gibbs state for a local Hamiltonian  $H = \sum_i H_i$ ,  $\|H_i\| \leq 1$ , care needs to be taken as to how to scale the energy of the Hamiltonian. On one hand, we would like the Gibbs state of a Hamiltonian  $H$  to be invariant under scaling of  $H$ , or perhaps rewriting  $H$  as a sum of possibly lower-rank projections. On the other hand, we note that it is unreasonable to expect to have a family of local Hamiltonians  $\{H_n\}_n$  with entanglement at room temperature (i.e. constant  $\beta > 0$ ), if the norm of  $H_n$  doesn't grow with the number of qubits  $n$ . Hence, we introduce the definition of energy density - which captures the average “energy” invested into a qubit in the system:

► **Definition 2 (Energy density).** A local Hamiltonian on  $n$  qubits with  $m$  local terms  $H = \sum_{i=1}^m H_i$ ,  $\|H_i\| \leq 1$  is said to have energy density  $\lambda = m/n$ .

The thermal Gibbs state is defined for a local Hamiltonian as follows:

► **Definition 3** (Gibbs state of a local Hamiltonian). *Let  $H = \sum_{i \in [m]} H_i$ , be a local Hamiltonian on  $n$  qubits  $\mathcal{H} = \mathbb{C}^{\otimes n}$ ,  $m$  local terms, and energy density  $\lambda = m/n$ . The Gibbs state of  $H$  for finite  $\beta > 0$  is the following density matrix:  $\frac{1}{Z}e^{-\beta \tilde{H}}$  where  $Z = \text{tr}(e^{-\beta \tilde{H}})$  and  $\tilde{H} = H/\lambda$ . For  $\beta \rightarrow \infty$  the Gibbs state is any  $\rho \in \ker(H)$ .*

#### Remark about the definition

One could also define the Gibbs state more strictly for  $T = 0$  or  $\beta \rightarrow \infty$  as the uniform mixture over codestates. Such a definition would not change the “quantumness” of the conjecture 1 nor the proof: the conjecture requires that at 0 temperature the system be topologically-ordered, so it must be highly entangled. As to the proof - it shows a circuit lower bound for every mixed ground-state, and in particular the uniform mixture thereof. The more relaxed definition here for  $T = 0$  is designed to include every state which is a stationary distribution of the Metropolis-Hastings algorithm, and indeed any mixed/pure ground-state is such a stationary distribution.

## 5 Preliminaries II :Quantum Error-Correcting Codes

We require the basic definition of stabilizer codes and CSS codes

► **Definition 4** (Quantum Stabilizer Code and Quantum CSS Code). *A stabilizer group  $\mathcal{G} \subseteq \mathcal{P}^n$  is an Abelian subgroup of  $\mathcal{P}^n$ . The codespace  $\mathcal{C}$  is then defined as the centralizer of  $\mathcal{G}$ , denoted by  $C[\mathcal{G}]$ , or equivalently - the mutual 1-eigenspace of  $\mathcal{G}$ . A CSS code  $\mathcal{C} = (\mathcal{C}_x, \mathcal{C}_z)$  is a stabilizer code where the check terms (i.e. generators of the group) are tensor-products of either only Pauli  $X$  or only Pauli  $Z$ . In particular regarding  $\mathcal{C}_x, \mathcal{C}_z$  as  $\mathbb{F}_2$  subspaces of  $\mathbb{F}_2^n$  we have  $\mathcal{C}_x \subseteq \mathcal{C}_z^\perp$  and vice versa.*

In this work, we will require some bounds on the minimal depth of a quantum circuit to generate a quantum code state. We recall a slight rephrasing of Prop. 45 in [5] to mixed states: <sup>2</sup>

► **Lemma 5** (Robust circuit lower bound for CSS code-states, [5]). *Let  $\mathcal{C}$  be a quantum CSS code of non-zero number of logical qubits  $k > 1$  on  $n$  qubits with minimal distance  $n^\epsilon$  for some  $\epsilon > 0$ . Let  $\rho_{gs}$  be a mixture on a set of code-states of  $\mathcal{C}$  and let  $V$  be a unitary circuit on  $N \geq n, N = \text{poly}(n)$  qubits that approximates  $\rho_{gs}$ :  $\rho = \text{tr}_T(V|0^{\otimes N}\rangle\langle 0^{\otimes N}|V^\dagger)$   $\|\rho - \rho_{gs}\|_1 \leq n^{-2}$ ,  $\rho_{gs} \in \mathcal{C}$  Then the depth of  $V$  is  $\Omega(\log(n))$ .*

### 5.1 Quantum Locally Testable Codes

In [2] Aharonov and the author defined quantum locally testable codes (qLTC’s). We state here a version due to Eldar and Harrow [5]: a quantum locally testable code can be defined by the property that quantum states on  $n$  qubits at distance  $d$  to the codespace have energy  $\Omega(d/n)$ .

<sup>2</sup> The only change required to achieve a mixed-state version of Prop. 45 is to construct distant subsets  $C_0, C_1$  of  $\mathbb{F}_2^n$  that each has a non-negligible probability, when measuring the ground-state  $\rho$  in one of the  $X/Z$  bases. Such sets are readily available: consider some eigenbasis of the code: In at least one of the  $Z$  or  $X$  bases the state  $\rho$  has support at least  $1/2$  on basis vectors such that each of them is a “well-partitioned” pure-state. One then constructs the sets  $C_0, C_1$  by “cutting in half” each such well-partitioned eigenstate, sending each half to a different set. By the minimal distance of the code, the “half” of each eigenstate is far from any “half” of any other eigenstate.

► **Definition 6.** If  $V$  is a subspace of  $(\mathbb{C}^2)^{\otimes n}$  then define its  $t$ -fattening to be  $V_t := \text{Span}\{(A_1 \otimes \cdots \otimes A_n)|\psi\rangle : |\psi\rangle \in V, \#\{i : A_i \neq I\} \leq t\}$ . Let  $\Pi_{V_t}$  project onto  $V_t$ . Then define the distance operator  $D_V := \sum_{t \geq 1} t(\Pi_{V_t} - \Pi_{V_{t-1}})$ .

This reflects the fact that for quantum states, Hamming distance should be thought of as an observable, meaning a Hermitian operator where a given state can be a superposition of eigenstates.

► **Definition 7** (Quantum locally testable code). An  $(q, s)$ -quantum locally testable code  $\mathcal{C} \subseteq (\mathbb{C}^2)^{\otimes n}$ , is a quantum code with  $q$ -local projection  $C_1, \dots, C_m$  such that  $\frac{1}{m} \sum_{i=1}^m C_i \succeq \frac{s}{n} D_{\mathcal{C}}$ .  $s$  is called the soundness parameter of the code.

We note that the soundness parameter  $s$  in this definition generalizes the standard notion of soundness of a classical LTC as a special case, where all  $C_i$ 's are diagonal in the computational basis. In particular, if the quantum code is a stabilizer code, then the definition of quantum local testability can be further simplified to resemble classical local testability more closely:

► **Definition 8** (Stabilizer Locally-Testable Codes (sLTC)). An sLTC is a quantum stabilizer code that is qLTC. An equivalent group-theoretic of an sLTC is as follows:  $\mathcal{C}$  is a stabilizer code generated by stabilizer group  $\mathcal{G}$ . It is  $(q, s)$ -sLTC if there exists a set  $S$  of  $q$ -local words in the stabilizer group  $g_1, \dots, g_t \in \mathcal{G}$  such that for  $P \in \mathcal{P}^n$  we have  $\mathbb{P}_{g \sim U[S]}([g, P] \neq 0) \geq (|P|/n) \cdot s$  where  $|P|$  is the Hamming weight of  $P$  modulo the centralizer of  $\mathcal{G}$ ,  $C[\mathcal{G}]$ :  $|P| = \min_{z \in C[\mathcal{G}]} \text{wt}(P + z)$  where for  $x \in \mathcal{P}^n$   $\text{wt}(x)$  counts the number of non-identity entries in  $x$ .

See [5] and [2] for the derivation of this definition as a special case of Definition 7: the operator  $D_{\mathcal{C}}$  penalizes a quantum state according to the “weighted” distance of that state from the codespace, whereas in definition 8 the penalty is defined w.r.t. each Pauli error separately, and as a function of the standard Hamming weight of the error, modulo the code.

Given a  $(q, s)$ -sLTC one can generate a sLTC with parameters  $(\lceil q/s \rceil, 1/e)$  by amplification as follows:

► **Proposition 9** (Randomized Amplification). Given is a  $(q, s)$  sLTC on  $n$  qubits with  $\text{poly}(n)$  checks. There exists a qLTC  $\mathcal{C}_{\text{amp}}$  of  $\text{poly}(n)$  checks with parameters  $(\lceil q/s \rceil, 1/e)$  where each qubit is incident on at most  $D' = \lceil q \log^2(n)/s \rceil$  checks.

## 6 Preliminaries III: Expansion of Small Errors on the Projective Hypercube

The main observation of this section is that while the projective code is a qLTC with a mild soundness parameter  $1/\log^2(n)$ , the soundness parameter for *small* errors is much better, and in fact for very small errors, their boundary (i.e. the Hamming weight of their image) is very close to maximal. We begin with a couple of standard definitions:

► **Definition 10** (Shadow). Let  $[n]^r$  denote the set of all  $r$ -subsets of  $[n]$ , and let  $\mathcal{A} \subseteq [n]^r$ . The lower shadow of  $\mathcal{A}$  is the set of all  $r-1$  subsets which are contained in at least one element of  $\mathcal{A}$ :  $\partial^- \mathcal{A} = \{A - \{i\} : A \in \mathcal{A}, i \in A\}$  and the upper shadow of  $\mathcal{A}$  is the set of all  $r+1$  subsets that contain at least one element of  $\mathcal{A}$ :  $\partial^+ \mathcal{A} = \{A + \{i\} : A \in \mathcal{A}, i \notin A\}$

Recall our notation that  $n$  denotes the dimension of the cube, and  $N = 2^n$  signifies the number of qubits in the context of the quantum error correcting code generated by the projective code of dimension  $n$ . We define  $p$ -faces as follows:

► **Definition 11** (*p*-face, set of *p*-faces, subspaces of *p*-faces). For integer  $n \geq 1$  a *p*-face is a word in  $\{0, 1, *\}^n$  that contains exactly *p* positions with \*. We denote by  $\mathcal{K}_p^N$  as the set of *p*-faces of the *n*-th cube. Let  $C_p^N$  denote the space spanned by  $\mathcal{K}_p^N$  with coefficients from  $\mathbb{F}_2$ .

One can think about a *p*-face as a subset of  $\{0, 1\}^n$  of all points that are equal to the *p*-face in its non-\* positions. Under this notation one can naturally define upper and lower shadow of *p*-faces as follows:

► **Definition 12** (Shadow of *p*-faces of  $\mathcal{K}_p^N$ ). The lower-shadow  $\partial^-$  of a *p*-face *f* is the set of all  $p - 1$  faces derived by replacing any \* entry with either 0 or 1. The upper-shadow  $\partial^+$  of a *p*-face *f* is the set of all  $p + 1$  faces that can be derived by replacing any non-\* entry of *p* with \*.

To connect the definitions above, note that the  $\mathbb{F}_2$ -boundary operator  $\partial_{p+1}$  associated with the  $\mathbb{F}_2$ -complex chain  $\{C_p^n\}_p$  maps each  $p + 1$ -face *f* to a summation over the set of *p*-faces  $\partial^- f$  with coefficient 1 in  $\mathbb{F}_2$ , whereas the co-boundary map  $\partial_p^T$  sends each  $p - 1$  face *f* to a summation over the set of *p*-faces  $\partial^+ f$  with coefficient 1.

Importantly, in this work, we will focus on the *p*-faces of the projective cube as the combinatorial set  $\mathcal{K}_p^N$ , and not on the corresponding  $\mathbb{F}_2$ -space  $C_p^N$ . This is because we are interested in establishing a combinatorial expansion property of the boundary maps  $\partial^+, \partial^-$ , to be later used in conjunction with the Sipser-Spielman decoder.

However, we will use, in a black box fashion, the properties of these maps, as maps over an  $\mathbb{F}_2$  complex chain that appeared in [16]: these properties are namely the soundness and minimal distance of a quantum code derived by the pair  $(\partial^+, \partial^-)$ .

In this study, we consider *p*-faces of the *n*-hypercube. While this resembles the case of subsets of  $[n]^r$  there is a major difference - since now any \*-entry replaced, can assume a value either 0 or 1, and the isoperimetric inequality needs to account for this larger set. Bollobas and Radcliffe provide an isoperimetric inequality for the regular grid [Thm. 10, Bollobas and Radcliffe]. Their bounds are useful especially when the set of faces is exponentially large in the dimension of the embedding space. For our purposes though, we are interested in set of *p*-faces that are polynomial in that dimension so simpler bounds are available as follows:

► **Lemma 13.** Let  $\mathcal{A} \subseteq \mathcal{K}_{p-1}^n$  be a set of  $(p - 1)$ -faces for  $p = n/2$ ,  $|\mathcal{A}| \leq n/32$ . Then  $|\partial^+ \mathcal{A}| \geq |\mathcal{A}| \cdot (n/2 + 1) \cdot (15/16)$ . Let  $\mathcal{A} \subseteq \mathcal{K}_{p+1}^n$  be a set of  $p + 1$ -faces for  $p = n/2$ ,  $|\mathcal{A}| \leq n/8$ . Then  $|\partial^- \mathcal{A}| \geq 2 \cdot |\mathcal{A}| \cdot (n/2 + 1) \cdot (15/16)$

## 6.1 The Projective Code

► **Definition 14** (The Projective Cube). Let  $\mathcal{K}_p^N$  denote the set of *p*-faces of the *N*-th cube. The projective cube, denoted by  $\tilde{\mathcal{K}}_p^N$  is formed by identifying  $x \sim \bar{x}$  iff  $x = \bar{x} + \mathbf{1}$ . Let  $\tilde{C}_p^N$  denote the space spanned by  $\tilde{\mathcal{K}}_p^N$  with coefficients in  $\mathbb{F}_2$ .

In this study, we will use build upon the projective code defined by Leverrier et al. [16]:

► **Definition 15** (Projective code). Extend the operators  $\partial^+, \partial^-$  from  $\mathcal{K}_p^N$  to  $\tilde{\mathcal{K}}_p^N$  and consider the complex chain formed by the  $\mathbb{F}_2$  span of  $\tilde{\mathcal{K}}_p^N$ , namely the spaces  $\{\tilde{C}_p^N\}_p$ :  $\tilde{C}_{p+1}^N \xrightarrow{\partial_{p+1}} \tilde{C}_p^N \xrightarrow{\partial_p} \tilde{C}_{p-1}^N$  the quantum CSS code (see Definition 4) defined by  $\mathcal{C}_x = \ker(\partial_p), \mathcal{C}_z = (\text{Im} \partial_p)^\perp$  is called the  $(N, p)$ -projective code and denoted by  $\mathcal{C}_{N,p} = (\mathcal{C}_x, \mathcal{C}_z)$ .

► **Lemma 16** (Properties of the projective code, [16]). For every sufficiently large *N* there exists  $n = 2^{\Omega(N)}$  such that the  $(N, p)$ -projective code  $\mathcal{C}_{N,p}$  for  $p = N/2$  has parameters  $[[n, 1, n^c]]$ , for some constant  $c > 0$ . It has soundness  $1/\log^2(n)$  and each qubit is incident on at most  $D = 2\log(n)$  checks.



We conclude this section by reducing the isoperimetric inequality for the projective cube to the isoperimetric inequality for the  $N$ -cube.

► **Lemma 17** (Isoperimetric inequalities for the projective hypercube). *Let  $\mathcal{C} = (\mathcal{C}_x, \mathcal{C}_z)$  denote the  $(N, p)$ -projective code with  $p = N/2$ . Let  $\mathcal{E}$  be a subset of errors of weight at most  $N/64$ . Then the number of checks  $\mathcal{C}_x$  incident on  $\mathcal{E}$  is at least  $|\mathcal{E}| \cdot (N/2) \cdot (15/16)$  and the number of  $\mathcal{C}_z$  checks incident is at least  $|\mathcal{E}| \cdot (N) \cdot (15/16)$*

### Some context

To provide some context, we note that at first sight it is unclear why considering such small weight ( $N/64$ ) may provide a non-trivial result: after all, for the regime of temperatures we are considering the typical error has nearly linear weight - i.e.  $n/\text{polylog}(n)$ , and since  $n = 2^N$  the weight considered above is merely  $\text{polylog}(n)$ . The reason is that as we later show in the proof, the typical error of the Gibbs state is not arbitrary, but can be further characterized as being formed on very small clusters - clusters of logarithmic size (see Lemma 27). We would like the check terms of the  $p$ -th projective code  $\mathcal{C}_{N,p}$  to be such that *any* error of logarithmic size expands very well in the Tanner graph of the code. The isoperimetric inequality provided here on this very restricted error model will allow us to argue that we can use a Sipser-Sipelman type decoder to correct all errors of the thermal state with high probability.

## 7 Behavior of Errors in the Gibbs State of qLTCs

### 7.1 The Thermal Gibbs by the Metropolis-Hastings Algorithm

As mentioned in the introduction, a recurring barrier in the emergent field of robust quantum entanglement is to establish a connection between the energy of a state, w.r.t. some local Hamiltonian, and the “error” experienced by that state.

The main observation in this section is that specifically for qLTC’s the Gibbs state can be formulated as a random error process (and specifically, a discrete finite Markov process) where the errors occur independently at each step, with an error rate that is comparable to the energy parameter of the state. This will then allow us to conclude that for sufficiently small energy of the Gibbs state the resulting errors can only form very small clusters.

► **Definition 18** (The Metropolis-Hastings Random Process Stabilizer Hamiltonians). *Let  $\mathcal{G}$  be a stabilizer group with a corresponding Hamiltonian  $H = H(\mathcal{G})$  on  $n$  qubits  $H = \sum_{i=1}^m H_i$  with  $m$  local terms, and  $\lambda(H) = \lambda = m/n$ . Let  $\beta \geq 0$  be finite. Define a Markov random process  $\mathcal{M}$  on a finite graph  $G = (V, E)$  whose vertex set  $V$  is formed by considering the uniform mixture  $\tau_0$  on the set of zero-eigenstates of  $H$ , and an additional vertex for each unique state formed by applying a Pauli error applied to  $\tau_0$ :  $V := \{P \cdot \tau_0 \cdot P, \quad P \in \mathcal{P}^n\}$ . For any two vertices  $\tau_i, \tau_j$  such that  $\tau_j = P\tau_i P$  where  $P$  is a single qubit Pauli  $P \in \mathcal{P}$  we define the following transition probabilities:  $\forall i \neq j \quad \mathcal{M}_{i,j} = \frac{1}{4n} \min \{1, \exp \{ \beta(E_{\tau_i} - E_{\tau_j}) / \lambda \} \}$  and  $\mathcal{M}_{i,i} = 1 - \sum_{j \neq i} \mathcal{M}_{i,j}$  where  $E(\tau_i) = \text{tr}(\tau_i H)$*

We note that under the definition above, any two vertices connected by an element of the stabilizer group  $g \in \mathcal{G}$ , i.e.  $\tau_i = g\tau_j g^\dagger$  will correspond to the same vertex - since it preserves the uniform distribution on the codespace. In particular we have  $|V| = |\mathbb{F}_2^n / \mathcal{C}_x| = |\mathbb{F}_2^n / \mathcal{C}_z|$ . More generally for stabilizer codes, each vertex corresponds to a minimal weight error modulo the stabilizer group. Also note, that the transition probabilities  $\mathcal{M}_{i,j}$  correspond to a 2-step process, where at the first step one samples a uniformly random index  $k \in [n]$  and then

applies a uniformly random Pauli error  $\mathcal{E}$  on that index with probability corresponding to the exponent of energy differences. We also note that the normalization by factor of  $4n$  stems from the size of the single-qubit Pauli group  $|\mathcal{P}| = 4$ .

► **Fact 19.** *There exists a stationary distribution of  $\mathcal{M}$ , denoted by  $\rho_0$  and it satisfies:  $\rho_0 = \frac{1}{Z} e^{-\beta \tilde{H}}$  where  $\tilde{H} = H/\lambda$  and  $Z = \text{tr}(e^{-\beta \tilde{H}})$  is the partition function for value  $\beta$ .*

## 7.2 The Thermal Gibbs Markov Process for qLTC's

As a next step, we consider a *truncated* random process  $\mathcal{M}_k$  for integer  $k$  where one only considers errors up to some “typical” weight  $k$ , beyond which the measure of the stationary distribution of the original process  $\mathcal{M}$  is negligible.

► **Definition 20** (*k*-Truncated Markov chain). *Let  $\mathcal{C}$  be a quantum stabilizer code on  $n$  qubits with  $m$  checks, and let  $H = H(\mathcal{C})$ . Set  $\lambda(H) = \lambda = m/n$ . Let  $\beta \geq 0$  be finite. For any two vertices  $\tau_i, \tau_j$  such that  $\tau_j = P\tau_i P$  where  $P$  is a single qubit Pauli  $P \in \mathcal{P}$  we define the following transition probabilities:*

$$\forall i \neq j \quad \mathcal{M}_{i,j} = \begin{cases} 0 & \text{if } \Delta(\tau_j, \tau_0) > k/n \\ \frac{1}{4n} \min \{1, \exp \{ \beta(E_{\tau_i} - E_{\tau_j})/\lambda \} \} & \text{o/w} \end{cases}$$

where  $\Delta(\tau_i, \tau_j)$  is the minimal weight of a Pauli  $P$  such that  $P\tau_i P = \tau_j$ , and  $\mathcal{M}_{i,i} = 1 - \sum_{i \neq j} \mathcal{M}_{i,j}$

In general, given the energy parameter  $\beta > 0$  one cannot bound a so-called “typical” weight, for which the measure of errors above that weight are negligible in the thermal Gibbs state  $e^{-\beta H}$ . However, for the specific case of qLTC's such a bound is readily available, via the soundness parameter  $\varepsilon > 0$ .

► **Proposition 21** (Truncated Metropolis Hastings Approximates the Gibbs State of a qLTC). *Suppose in particular that  $H = H(\mathcal{C})$  where  $\mathcal{C}$  is a  $(q, s)$  sLTC, and set  $\lambda = \lambda(H)$ . Let  $0 < \delta < 1/2$  and denote  $k = n\delta$ . Let  $\rho_k$  denote a stationary distribution of the  $k$ -th truncated Markov chain  $\mathcal{M}_k$ . Then for  $\beta \geq 5 \ln(1/\delta)/s$  the  $k$ -th truncated Markov chain approximates the thermal Gibbs state of the scaled Hamiltonian  $\tilde{H} = H/\lambda$ :  $\left\| \rho_k - \frac{1}{Z} e^{-\beta \tilde{H}} \right\| \leq 2n \cdot e^{-2n \cdot \ln(1/\delta) \cdot \delta}$ ,  $Z = \text{tr}(e^{-\beta \tilde{H}})$*

## 7.3 Percolation Behavior of Random Errors in the Gibbs State of qLTC's

We now recall some of the definitions of Fawzi et al. [6]. The first one is that of an  $\alpha$ -subset which is a subset that has a large intersection with some fixed subset:

► **Definition 22** ( $\alpha$ -subset). *Let  $G = (V, E)$  be a graph,  $X \subseteq V$ , and  $\alpha \in [0, 1]$ . An  $\alpha$ -subset of  $X$  is a set  $S \subseteq V$  such that  $|S \cap X| \geq \alpha \cdot |S|$ . We denote by  $\text{maxconn}_\alpha(X)$  as the maximum size of an  $\alpha$  connected subset of  $X$ .*

The second definition is that of a locally-stochastic random error model, which generalizes an independent random error model in that the probability of a set decays exponentially in its size:

► **Definition 23** (Locally-stochastic). *Let  $V$  be a set of  $n$  elements. A random subset  $X \subseteq V$  is said to be locally-stochastic with parameter  $p \in [0, 1]$  if for every  $S \subseteq V$  we have  $P(X \supseteq S) \leq p^{|S|}$*

We now recall Theorem 17 of [6] on the percolation behavior of  $\alpha$ -subsets. It states, roughly, that the size of the maximal  $\alpha$ -connected component when choosing vertices at random with probability  $p$  drops exponentially in  $dp^\alpha$ . We rephrase the theorem as the following lemma:

► **Lemma 24** (Percolation behavior for locally-stochastic random errors). *Let  $G = (V, E)$  be a graph on  $n$  vertices, such that each vertex has at most  $D = D(n)$  neighboring edges. Let  $\alpha > 0$ . Let  $X \subseteq V$  be a random subset of  $V$  that is locally stochastic with parameter  $p$ . There exists a constant  $c$  such that if  $p < c/D$  we have  $\mathbb{P}(\text{maxconn}_\alpha(X) \geq t) \leq 2n \cdot (2Dep^\alpha)^t$*

Consider now a local Hamiltonian  $H$ , we define its interaction graph as follows:

► **Definition 25** (Interaction graph of a local Hamiltonian). *Let  $H = \sum_i H_i$  denote a local Hamiltonian on  $n$  qubits. The interaction graph of  $H$ ,  $G(H) = (V, E)$  is defined by  $V = [n]$  corresponding to the  $n$  qubits, and  $e = (i, j) \in E$  if qubits  $i$  and  $j$  share a local term  $H_e$  in  $H$ .*

We would like to show that the  $k$ -th truncated Metropolis-Hastings random process on  $H$  is locally-stochastic for sufficiently small  $k$ .

To see why this is a non-trivial statement, recall that the MH random process does not induce independent errors, since the probability of adding error to a given qubit depends on the additional energy cost induced by flipping that qubit, and that additional energy depends on the specific error configuration on its neighboring qubits.

In fact this random error model implies that errors are *more likely* to occur near previously sampled errors thus leading to a behavior that is completely opposite to local stochasticity. However, we show that if  $k$  is significantly less than  $n/D$  then this effect is negligible compared to the probability of sampling an error that is not connected to any other error, and hence approximately these errors are locally-stochastic.

► **Lemma 26** (The Thermal Gibbs State is Locally-Stochastic). *Let  $\mathcal{C}$  be a stabilizer code and let  $H = H(\mathcal{C})$  denote the corresponding local Hamiltonian. Suppose that the corresponding interaction graph  $G(H)$  has degree at most  $D$ . Let  $\alpha \in (0, 1]$ , and consider the  $k$ -th truncated Markov chain  $\mathcal{M}_k$  and its stationary distribution  $\rho_k$ , for  $k \leq \frac{n}{2e(De^{300})^{1/\alpha}}$ . If the energy density is sufficiently large compared to the inverse temperature:  $\lambda \geq \beta \ln(n)$  then  $\mathcal{E} \sim \rho_k$  is locally-stochastic with parameter at most  $p_0 \leq 2ke/n$  with probability at least  $1 - (k+1)n^{-4}$ .*

We conclude our central lemma of this section - which is that the thermal Gibbs state  $e^{-\beta H}$  where  $H$  is a Hamiltonian corresponding to a qLTC, and  $\beta$  is sufficiently large, satisfies a percolation property - namely that the maximal  $\alpha$ -connected component of a typical error  $\mathcal{E}$  is of logarithmic size:

► **Lemma 27** (Typical error components are small for the thermal state of qLTC's). *Let  $\mathcal{C}$  be a  $(q, s)$ -sLTC on  $n$  qubits and let  $H(\mathcal{C})$  be its corresponding Hamiltonian,  $\lambda(H) = \lambda$ . Suppose that the interaction graph of  $H$ ,  $G(H)$ , is of degree most  $D$ . Let  $\alpha > 0$ . Let  $\tau = \mathcal{E} \cdot \tau_0 \cdot \mathcal{E}$  be a random state (the uniform code mixed state conjugated by a random error  $\mathcal{E}$ ) sampled according to the distribution  $e^{-\beta H/\lambda}/Z$  for  $(10/\alpha) \cdot \ln(D)/s \leq \beta \leq \lambda/\ln(n)$ . Then  $\mathbb{P}(\text{maxconn}_\alpha \mathcal{E} > \ln(n)/100) \leq n^{-3}$*

### The range of values $\beta$

It is insightful at this point to consider the statement of the lemma w.r.t. the parameter  $\beta$ : the statement of the lemma requires that  $\beta$  is within some range - between  $\log\log(n)$  and  $\log(n)$ . This initially might seem strange as intuitively, increasing  $\beta$  can only improve the ability to correct errors since it corresponds to a regime of much fewer errors - e.g. lower temperature.

However in Lemma 26 it turns out that the analysis is more subtle: indeed we require  $\beta$  to be also sufficiently small so that the error model is locally stochastic: if  $\beta$  is too large (i.e. the temperature is very low) it turns out that a qubit that is hit by an error is much more likely to be hit by another error - this contrary to local stochasticity, whereas for higher temperatures this phenomenon is greatly alleviated.

Hence, the phenomenon of locally stochastic errors, that we exploit to demonstrate a shallow decoder is in fact relevant only for a median range of temperatures: for very low temperatures, the error is no longer locally stochastic, but in that range - the *absolute* number of errors is extremely small to allow worst-case error correction. For higher temperatures, the absolute number of errors is very large but conforms to the locally stochastic model which is treatable by a local decoder. This results in a “win-win” situation, which is handled case-by-case in the proof of the main theorem.

## 8 A Shallow Decoder for Low Error Rate

The last component of the proof is to demonstrate a shallow circuit that can correct the thermal state  $e^{-\beta H}$  to a code-state, for sufficiently large  $\beta > 0$  (finite or not). In the previous section we’ve seen that such a state can be modeled as a random error process with small rate. We would now like to leverage that understanding, together with the small-set expansion property of the  $n$ -projective cube to show that the quantum version of the Sipser Spielman decoder yields a shallow decoder.

Inspired by the decoding algorithm of Fawzi et al. [6] we propose an algorithm for decoding a random error  $\mathcal{E}$  in depth proportional to  $\log(\max\text{conn}_\alpha(\mathcal{E}))$ . It is based on a parallel version of the Sipser-Spielman decoder, which we state as follows:

► **Lemma 28** ([18], Theorem 11). (Parallel decoder for small-set expander graphs). *Let  $C$  be a code on  $n$  bits and let  $G$  denote the Tanner graph of  $C$ . Suppose  $G$  is a  $(c, d)$ -bi-regular graph on  $n$  vertices. The parallel decoder  $\mathcal{A}$  is an algorithm that given error  $\mathcal{E} = \mathcal{E}_1$  iteratively replaces it with errors  $\mathcal{E}_i$  for  $i \geq 1$ . At step  $i$  the algorithm may modify bits only in the support of  $\mathcal{E}_i \cup \Gamma(\mathcal{E}_i)$ , and in particular, examines for each bit  $k$  only  $\Gamma(k)$ . If, in addition, at the beginning of iteration  $i$  we have:  $|\Gamma(\mathcal{E}_i)| \geq |\mathcal{E}_i| \cdot c \cdot (3/4 + \varepsilon)$  for some constant  $\varepsilon > 0$ , then after step  $i$  the weight of the residual error  $\mathcal{E}_i$  decreases by a multiplicative factor:  $|\mathcal{E}_{i+1}| \leq |\mathcal{E}_i| \cdot (1 - 4\varepsilon)$*

Our quantum decoder is an application of the Sipser-Spielman decoder on the individual  $X, Z$  errors.

► **Algorithm 29** (Shallow Decoder  $\mathcal{B}$ ).

*Input: a quantum state  $\rho$  on  $n$  qubits, a set of  $X$  checks  $\mathcal{C}_x$  and a set of  $Z$  checks  $\mathcal{C}_z$ .*

1. *Run the decoder  $\mathcal{A}$  w.r.t.  $Z$  errors using  $\mathcal{C}_x$ .*
2. *Run the decoder  $\mathcal{A}$  w.r.t  $X$  errors using  $\mathcal{C}_z$ .*

► **Lemma 30.** *Consider the projective code  $\mathcal{C} = (\mathcal{C}_x, \mathcal{C}_z)$  on  $n$  qubits with  $p = n/2$ , and let  $\mathcal{E} \in \mathcal{P}^n$  denote an error with far-away and small connected components:  $\max\text{conn}_\alpha(\mathcal{E}) \leq \ln(n)/100$ ,  $\alpha = 1/(\gamma \log \log(n))$  where  $\gamma = \log(1 - 4 \cdot (3/16))$  is the constant implied by Lemma 28 for  $\varepsilon = 3/16$ . Then shallow decoder  $\mathcal{B}$  runs in depth at most  $2\gamma \log^2 \log(n)$  steps and satisfies:  $\mathcal{B} \circ \mathcal{E} \circ \rho = \rho \quad \forall \rho \in \mathcal{C}$*

We note here that the decoder  $\mathcal{B}$  requires extra ancillary bits for syndrome computation, hence the notation  $\mathcal{B} \circ \mathcal{E} \circ \rho$  signifies a quantum channel, where some of the qubits are discarded after computation.

## 9 Global Entanglement for Thermal States

### 9.1 The construction

#### 1. Step 1 – The projective code:

Fix  $n$  as the number of qubits in the code. As the basis for our construction we consider the  $(N, p)$  projective code  $\mathcal{C}$  for  $p = N/2$ . By Lemma 16 we can choose  $N = \Theta(\log(n))$  such that  $\mathcal{C}$  is a qLTC  $[[n, 1, n^c]]$  for some  $c > 0$  with qLTC parameters  $(q = \log(n), s = 1/\log^2(n))$ . By construction, the interaction graph of  $H(\mathcal{C})$ , i.e.  $G(H(\mathcal{C}))$  is  $D$ -regular with  $D = 2 \cdot \log(n)$  and the local Hamiltonian  $H$  has  $m = 2n \cdot \log(n)$  check terms.

#### 2. Step 2 – Amplification:

We apply Proposition 9 to conclude the existence of a qLTC, denoted by  $\mathcal{C}'$  with parameters  $(q' = \lceil \log^3(n) \rceil, s' = 1/e)$  and  $\lambda = \log^4(n)$ . The interaction graph of the Hamiltonian of  $\mathcal{C}'$ , i.e.  $G(H(\mathcal{C}'))$  has degree at most  $D' \leq \lceil \log^7(n) \rceil$ .

#### 3. Step 3 – Union:

Finally, we consider the union of the checks of  $\mathcal{C}$  and  $\mathcal{C}'$  and denote the union by  $\mathcal{C}_{pa}$  - this is our construction. We denote the number of checks by  $m_{pa}$ . We have that,  $\mathcal{C}_{pa}$  is  $[[n, 1, n^c]]$  quantum code, and is qLTC with parameters:  $(q_{pa} = \log^3(n), s_{pa} = 1/2e, D_{pa} \leq 2\log^7(n))$  and  $\lambda_{pa} \geq 2\log^4(n)$ .

We note that the amplified code  $\mathcal{C}'$  has constant soundness for all *non-zero* distances, but it is not clear a-priori why it should also satisfy  $\ker(\mathcal{C}) = \ker(\mathcal{C}')$ . Hence, the union of  $\mathcal{C}$  and  $\mathcal{C}'$  is taken in order to enforce the ground-state of the final code to equal that of  $\mathcal{C}_{pa}$ . This slightly reduces the soundness, and increases the degree of the interaction graph of the final code. Also note that the check terms of  $\mathcal{C}_{pa}$  commute in pairs.

### 9.2 Main Theorem

We now state formally our main theorem.

► **Theorem 31.** *Let  $\mathcal{C}_{pa}$  denote the code constructed above on  $n$  qubits, and let  $H = H(\mathcal{C}_{pa})$ ,  $\lambda = \lambda(H)$  and inverse temperature:  $\beta \geq 20e \cdot \log^2 \log(n)$ . Any quantum circuit  $U$  on  $a \geq n$  qubits that approximates the thermal state of  $\tilde{H} = H/\lambda$  on a set of qubits  $S$ ,  $|S| = n$ , at inverse temperature  $\beta$ ,  $\left\| \text{tr}_{-S}(U|0^{\otimes a}\rangle\langle 0^{\otimes a}|U^\dagger) - e^{-\beta \tilde{H}}/Z \right\|_1 \leq 0.1n^{-2}$  has depth at least  $d(U) = \Omega(\log(n))$*

---

#### References

- 1 Dorit Aharonov and Michael Ben-Or. Fault-tolerant quantum computation with constant error rate. *SIAM J. Comput.*, 38(4):1207–1282, July 2008.
- 2 Dorit Aharonov and Lior Eldar. Quantum locally testable codes. *SIAM J. Comput.*, 44:1230–1262, 2015.
- 3 R. Alicki, Michał Horodecki, P. Horodecki, and R. Horodecki. On thermal stability of topological qubit in kitaev's 4d model. *Open Systems and Information Dynamics*, 17, November 2008.
- 4 Robert Alicki, Michał Horodecki, Paweł Horodecki, and Ryszard Horodecki. Dynamical description of quantum computing: Generic nonlocality of quantum noise. *Phys. Rev. A*, 65:062101, May 2002.
- 5 Lior Eldar and Aram Wettroth Harrow. Local hamiltonians whose ground states are hard to approximate. *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 427–438, 2015.

- 6 Omar Fawzi, Antoine Grosse, and Anthony Leverrier. Efficient decoding of random errors for quantum expander codes. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2018, pages 521–534, New York, NY, USA, 2018. ACM.
- 7 Michael H. Freedman and Matthew B. Hastings. Quantum systems on non-k-hyperfinite complexes: A generalization of classical statistical mechanics on expander graphs. *Quantum Info. Comput.*, 14(1-2):144–180, January 2014.
- 8 Michael H. Freedman, Alexei Kitaev, Michael J. Larsen, and Zhenghan Wang. Topological quantum computation. *Bull. Amer. Math. Soc. (N.S.)*, pages 31–38, 2002.
- 9 Elizabeth Gibney. Inside microsofts quest for a topological quantum computer. *Nature*, 2016.
- 10 Daniel Gottesman. Fault-tolerant quantum computation with constant overhead. *Quantum Info. Comput.*, 14(15-16):13380–1372, November 2014.
- 11 Matthew B. Hastings. Topological order at nonzero temperature. *Phys. Rev. Lett.*, 107:210501, November 2011.
- 12 Matthew B. Hastings. Quantum codes from high-dimensional manifolds. In *ITCS*, 2017.
- 13 Matthew B. Hastings, Grant H. Watson, and Roger G. Melko. Self-correcting quantum memories beyond the percolation threshold. *Phys. Rev. Lett.*, 112:070501, February 2014.
- 14 A.Yu. Kitaev. Fault-tolerant quantum computation by anyons. *Annals of Physics*, 303(1):2–30, 2003.
- 15 A. Leverrier, J. Tillich, and G. Zémor. Quantum expander codes. In *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, pages 810–824, October 2015.
- 16 Anthony Leverrier, Vivien Londe, and Gilles Zémor. A construction of quantum (almost) locally testable codes. *Quantum Information Processing (QIP)*, 2019.
- 17 Shachar Lovett and Emanuele Viola. Bounded-depth circuits cannot sample good codes. In *IEEE Conference on Computational Complexity*, 2011.
- 18 M. Sipser and D. A. Spielman. Expander codes. *IEEE Transactions on Information Theory*, 42(6):1710–1722, November 1996.
- 19 Xiao-Gang Wen. Topological order: From long-range entangled quantum matter to a unified origin of light and electrons. *ISRN Condensed Matter Physics*, 2013, October 2012. doi: 10.1155/2013/198710.
- 20 Beni Yoshida. Feasibility of self-correcting quantum memory and thermal stability of topological order. *Annals of Physics*, 326(10):2566–2633, 2011.

# Time-Space Lower Bounds for Simulating Proof Systems with Quantum and Randomized Verifiers

Abhijit S. Mudigonda 

Portland, OR, USA

<https://abhijit-mudigonda.github.io/math/>

abhijitm@mit.edu

R. Ryan Williams 

EECS and CSAIL, MIT, Cambridge, MA, USA

<https://people.csail.mit.edu/rrw/>

rrw@mit.edu

---

## Abstract

A line of work initiated by Fortnow in 1997 has proven model-independent time-space lower bounds for the SAT problem and related problems within the polynomial-time hierarchy. For example, for the SAT problem, the state-of-the-art is that the problem cannot be solved by random-access machines in  $n^c$  time and  $n^{o(1)}$  space simultaneously for  $c < 2 \cos(\frac{\pi}{7}) \approx 1.801$ .

We extend this lower bound approach to the quantum and randomized domains. Combining Grover's algorithm with components from SAT time-space lower bounds, we show that there are problems verifiable in  $O(n)$  time with quantum Merlin-Arthur protocols that cannot be solved in  $n^c$  time and  $n^{o(1)}$  space simultaneously for  $c < \frac{3+\sqrt{3}}{2} \approx 2.366$ , a super-quadratic time lower bound. This result and the prior work on SAT can both be viewed as consequences of a more general formula for time lower bounds against small-space algorithms, whose asymptotics we study in full.

We also show lower bounds against randomized algorithms: there are problems verifiable in  $O(n)$  time with (classical) Merlin-Arthur protocols that cannot be solved in  $n^c$  randomized time and  $O(\log n)$  space simultaneously for  $c < 1.465$ , improving a result of Diehl. For quantum Merlin-Arthur protocols, the lower bound in this setting can be improved to  $c < 1.5$ .

**2012 ACM Subject Classification** Theory of computation → Complexity classes

**Keywords and phrases** Time-space tradeoffs, lower bounds, QMA

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.50

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/2012.00330>.

**Funding** *R. Ryan Williams*: Supported by NSF CCF-1909429 and CCF-1741615.

**Acknowledgements** We thank the anonymous reviewers for helpful comments. The first author thanks Ryan Williams for his support and patience throughout this research. The first author also thanks Aram Harrow, Peter Shor, Saeed Mehraban, Ashwin Sah, and Lisa Yang for contributing office space and helpful conversations and Lijie Chen and Shyan Akmal for reading and editing a draft of this manuscript. Lastly, the first author apologizes to the Theory Group at MIT CSAIL eating so many of the chocolate-covered pretzels.

## 1 Introduction

A flagship problem in computational complexity is to prove lower bounds for the SAT problem. While it is conjectured that no polynomial-time algorithms exist for SAT (in other words,  $P \neq NP$ ), not much progress has been made in that direction. Furthermore, several significant barriers towards such a separation are known [3, 22, 1]. Therefore, approaches have centered around proving weaker lower bounds on SAT first.



© Abhijit S. Mudigonda and R. Ryan Williams;  
licensed under Creative Commons License CC-BY  
12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 50; pp. 50:1–50:20



Leibniz International Proceedings in Informatics

LIPIcs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



A natural preliminary step in showing that no polynomial-time algorithm can decide SAT is showing that no algorithms of logarithmic space can decide SAT, or in other words, showing that  $L \neq NP$ . Unlike the P vs. NP problem, the aforementioned complexity barriers (arguably) do not apply as readily to L vs. NP, and concrete progress has been made.<sup>1</sup>

Following a line of work [11], R. Williams [25] proved that SAT (equivalently<sup>2</sup> NTIME[ $n$ ], nondeterministic linear time) cannot be decided by algorithms (even with constant-time random access to their input and storage) using both  $n^{o(1)}$  space and  $n^c$  time, for  $c < 2 \cos(\frac{\pi}{7}) \approx 1.802$ . If one could show the same lower bound for arbitrarily large constant  $c$ , the separation  $L \neq NP$  would follow immediately. In the following we use  $TS[n^c]$  to denote the class of languages decidable by  $n^{o(1)}$ -space algorithms using  $n^c$  time.

All the aforementioned work builds on the *alternation-trading proofs* approach [27]. This approach combines two elements: a *speedup rule* that reduces the runtime of an algorithm by “adding a quantifier” ( $\exists$  or  $\forall$ ) to an alternating algorithm, and a *slowdown rule* that uses a complexity theoretic assumption (for example,  $SAT \in TS[n^c]$ ) to “remove a quantifier” and slightly increase the runtime of the resulting algorithm. Both rules yield inclusions of complexity classes. Our ultimate goal is to contradict a time hierarchy theorem (e.g., proving  $n^{100}$  time computations can be simulated in  $n^{99}$  time) by applying these rules in a nice order, and with appropriately chosen parameters.

One may hope that the constant  $c$  from [25] can be made arbitrarily large, and eventually show that  $L \neq NP$ . Unfortunately, in [7], R. Williams and S. Buss showed that no alternation-trading proof based purely on the speedup and slowdown rules from that line of work could improve on the exponent of [25].

Nevertheless, there is hope that alternation-trading proofs might yield stronger lower bounds for problems harder than SAT. For example, R. Williams [27] showed that the  $\Sigma_2P$ -complete problem  $\Sigma_2SAT$  is not in  $TS[n^c]$ , for  $c < 2.903$ . In this paper, we make further progress in this direction. In particular, we focus on the quantum and randomized analogues of NTIME[ $n$ ], QCMATIME[ $n$ ] and MATIME[ $n$ ], obtaining stronger lower bounds against both classes.<sup>3</sup> We believe our lower bound for QCMATIME[ $n$ ] (Main Theorem 2) to be particularly interesting because it yields a *nontrivial* separation between a quantum complexity class and a classical complexity class *without the need for oracles*.<sup>4</sup> While there are several results [6, 21, 24] demonstrating the power of quantum computation against very restricted low-depth classical circuit models ( $NC^0$ ,  $AC^0$ ,  $AC^0[2]$ ) which also imply strong oracle separation results, our result appears to be the first non-trivial lower bound for a quantum class against the much more general random-access machine model (with simultaneous time and space constraints).

## 1.1 Our Results

### 1.1.1 Generic slowdown rules and a lower bound for QCMATIME[ $n$ ]

For showing stronger lower bounds on QCMATIME[ $n$ ], our key observation is that the stronger assumption  $QCMATIME[n] \subseteq TS[n^c]$  (compared to  $NTIME[n] \subseteq TS[n^c]$ ) can be applied to construct a stronger conditional slowdown rule. Formally, we generalize the

<sup>1</sup> For example, there exist oracles relative to which the lower bounds in the following paragraph are false.

<sup>2</sup> At least, up to polylogarithmic factors.

<sup>3</sup> Recall that QCMA (quantum classical Merlin-Arthur) is essentially NP with a quantum verifier and MA (Merlin-Arthur) is essentially NP with a randomized verifier.

<sup>4</sup> By “nontrivial”, we mean a separation that does not immediately follow from known classical results. For example,  $QCMATIME[n] \not\subseteq TS[n^{1.8}]$  follows immediately from the classical lower bound  $NTIME[n] \not\subseteq TS[n^{1.8}]$ , but our result does not.

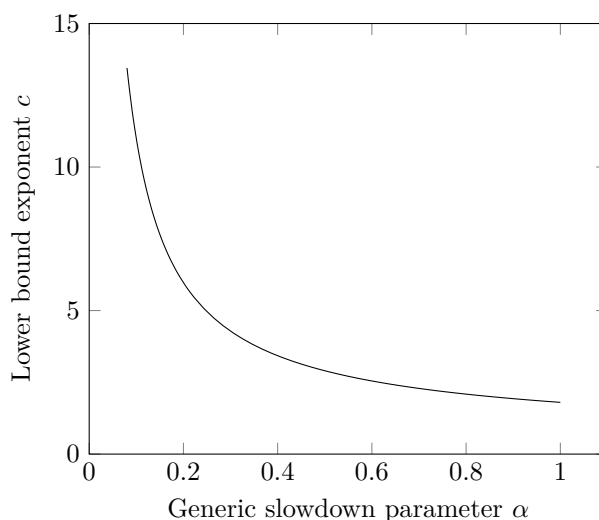
previous framework of alternation-trading proofs by introducing the notion of a *generic slowdown rule* (defined formally in Definition 11), which are slowdown rules parameterized by a constant  $\alpha \in (0, 1]$  controlling the runtime cost associated with removing a quantifier. Smaller values of  $\alpha$  correspond to stronger slowdown rules. We prove the following theorem showing how generic slowdown rules imply time-space lower bounds.

► **Main Theorem 1.** *Fix  $\alpha \in (0, 1]$  and let  $\mathcal{C}$  be a complexity class. Let  $r_1$  be the largest root of the polynomial  $P_\alpha(x) := \alpha^2 x^3 - \alpha x^2 - 2\alpha x + 1$ . If  $\mathcal{C} \subseteq \text{TS}[n^c]$  implies a generic slowdown rule with parameters  $\alpha$  and  $c$ , then  $\mathcal{C} \not\subseteq \text{TS}[n^c]$  for  $c < r_1$ .*

The assumption  $\text{NTIME}[n] \subseteq \text{TS}[n^c]$  implies a slowdown rule with  $\alpha = 1$ . The previous  $\text{NTIME}[n] \not\subseteq \text{TS}[n^c]$  for  $c < 2 \cos(\frac{\pi}{7})$  lower bound [25] becomes an immediate corollary of Main Theorem 1 if we take  $\mathcal{C} = \text{NTIME}[n]$ . The stronger slowdown rule that we obtain from the stronger assumption  $\text{QCMATIME}[n] \subseteq \text{TS}[n^c]$  has  $\alpha = \frac{2}{3}$ , which allows us to derive lower bounds for larger values of  $c$ . In particular, we obtain the following lower bound for Quantum (Classical) Merlin-Arthur linear time.

► **Main Theorem 2.**  $\text{QCMATIME}[n] \not\subseteq \text{TS}[n^c]$  for  $c < \frac{3+\sqrt{3}}{2} \approx 2.366$ .

The main advantage of the generic slowdown approach and Main Theorem 1 is that improvements in slowdown rules translate immediately into stronger bounds against TS. Figure 1 shows how the lower bound exponent we obtain changes as  $\alpha$  does. As expected, the lower bound exponent goes to infinity as  $\alpha$  approaches zero, but we see that even modest improvements in  $\alpha$  yield substantially stronger bounds. We discuss potential applications further in Section 1.3. In the full version of the paper, we show that this dependence between the generic slowdown parameter and lower bound exponent is “optimal” for the tools we use, extending the optimality theorem of Buss and Williams [7] to the general case while also providing a shorter proof of their optimality theorem.



■ **Figure 1** The lower bound exponent as a function of the generic slowdown parameter  $\alpha$ .

### 1.1.2 Lower bounds against randomized logspace

We also prove lower bounds against randomized logspace with two-sided error. While the techniques used in this setting are similar, the results do not follow from Main Theorem 1, so we state them separately. When discussing the randomized setting, we will slightly abuse

notation by referring to the class of languages decidable by a logspace machine in  $n^c$  time as  $\text{BPTS}[n^c]$ .<sup>5</sup> We prove lower bounds for both linear-time Merlin-Arthur protocols and linear-time Classical-Merlin Quantum-Arthur protocols.

► **Main Theorem 3.** *Let  $r_1 \approx 1.465$  be the largest root of the polynomial  $x^3 - x^2 - 1$ . Then,  $\text{MATIME}[n] \not\subseteq \text{BPTS}[n^c]$  for  $c < r_1$ . Furthermore,  $\text{QCMATIME}[n] \not\subseteq \text{BPTS}[n^c]$  for  $c < 1.5$ .*

Prior to this work, the state-of-the-art, due to [10, 8], was that  $\text{MATIME}[n] \not\subseteq \text{BPTS}[n^c]$  for  $c < \sqrt{2} \approx 1.414$ . Observe that  $\text{MATIME}[t] \subseteq \text{PPTIME}[t^2]$  because we can amplify Arthur’s completeness and soundness to  $(1 - 2^{-100t}, 2^{-100t})$  while increasing the runtime of the verifier by a factor of  $O(t)$ , and we can union bound over Merlin’s strings.<sup>6</sup> A similar argument, coupled with the quasilinear-time simulation of bounded-error quantum computation with unbounded-error random computation of [23], shows that  $\text{QCMATIME}[t] \subseteq \text{PPTIME}[t^2]$ . Therefore, pushing either of the lower bound exponents in Main Theorem 3 to beyond 2 would yield superlinear bounds for decision versions of counting-type problems (e.g. MAJ-SAT) against randomized logspace. (Note it is known that  $\#\text{SAT}$  requires  $\tilde{O}(n^2)$  time for randomized logspace [17].) While these results are admittedly incremental improvements, they use some different ideas compared to previous works, and may be amenable to further improvement (see Section 1.3 for more details).

The lower bounds of Main Theorem 2 and Main Theorem 3 actually hold for complexity classes that are presumably even “smaller” than  $\text{QCMATIME}[n]$  and  $\text{MATIME}[n]$ ; we describe these further in Section 2.

## 1.2 Techniques

### 1.2.1 Alternation-trading proofs

Many time-space lower bounds for SAT and related problems are proved via *alternation-trading proofs*, which give a chain of inclusions of complexity classes. We will formally define alternation-trading proofs in Section 2; for now, let us give a cursory explanation. An *alternation-trading proof* consists of a sequence of containments of *alternating complexity classes*. An alternating complexity class can be thought of as a “fine-grained” version of  $\Sigma_k\text{P}$  or  $\Pi_k\text{P}$ : it is a complexity class parameterized by  $(k + 1)$  positive constants bounding the length of the output of each quantifier and the verifier runtime. For example,

$$(\exists n^2)(\forall n^2)\text{TS}[n^5]$$

is an alternating complexity class. This notation refers to the class of languages decided by a  $\Sigma_2$  machine where, on inputs of length  $n$ , the two quantifiers each quantify over  $\tilde{O}(n^2)$  bit strings and the *verifier runtime* is  $\tilde{O}(n^5)$ .

In an alternation-trading proof, there are two main ways of passing from one alternating complexity class to the next. The first is a **speedup rule**, which adds a quantifier to the class, and decreases the verifier runtime. For example, a speedup rule might yield an inclusion of the form

$$\dots \text{TS}[n^d] \subseteq \dots (Qn^x)(\neg Qx \log n) \text{TS}[n^{d-x}] \quad (1)$$

<sup>5</sup> We call this a “slight abuse” because we defined  $\text{TS}$  to be a class of languages decided using  $n^{o(1)}$  space, whereas  $\text{BPTS}$  is defined with respect to  $O(\log n)$  space.

<sup>6</sup> By a union bound, there is a gap between the case where all  $2^t$  Merlin strings have a  $2^{-100t}$  chance of accepting, and the case where a single Merlin string accepts with probability at least  $1 - 2^{-100t}$ .

for some constant  $0 < x < d$  and quantifier  $Q \in \{\exists, \forall\}$ , where  $\neg Q$  denotes the opposite quantifier and the  $\dots$  refer to other quantifiers. Two important points to note are that (a) the speedup rule is generally an unconditionally true inclusion and (b) the second quantifier has only  $O(\log n)$  bits.

The second major component of alternation-trading proofs is a **slowdown rule**, which removes a quantifier and increases the verifier runtime. We will use slowdown rules that hold conditioned on complexity-theoretic assumptions (that we will later contradict). For example, the slowdown rule used to prove lower bounds on  $\text{NTIME}[n]$  can be informally stated as follows: assuming  $\text{NTIME}[n] \subseteq \text{TS}[n^c]$  for some  $c > 0$ ,

$$\dots(Qn^a)(\neg Qn^b)\text{TS}[n^d] \subseteq \dots(Qn^a)\text{TS}[n^{c \cdot \max(a,b,d)}]. \quad (2)$$

where again  $Q \in \{\exists, \forall\}$  and  $\neg Q$  denotes the opposite quantifier. This rule follows from an application of padding/translation.

While the speedup and slowdown rules are themselves simple, the existing lower bounds on  $\text{NTIME}[n]$  arise by applying these rules in a long intricate sequence, with appropriately chosen parameters for the speedup rule applications. Ultimately, we aim to use slowdown and speedup rules to exhibit a sequence of inclusions that shows (for example) that  $\text{NTIME}[n^d] \subseteq \text{NTIME}[n^{d'}]$  for  $d' < d$ , contradicting the nondeterministic time hierarchy theorem and demonstrating that our initial assumption must have been false.

All time-space lower bounds for SAT against random-access models of computation, including the state-of-the-art bound [25], use an alternation-trading proof. This particular proof will be a starting point for this work.

### 1.2.2 Generic slowdown rules

We start by introducing the notion of a generic slowdown rule. Generic slowdown rules are parameterized by a constant  $\alpha$  such that  $0 < \alpha \leq 1$ , along with a constant  $c \geq 1$  that generally comes with an assumption being made. Informally, generic slowdown rules allow us to – under an appropriate assumption – prove conditional inclusions of the form

$$\dots(Qn^a)(\neg Qn^b)\text{TS}[n^d] \subseteq \dots(Qn^a)\text{TS}[n^{\alpha c \cdot \max(a,b,d)}].$$

Observe that when  $\alpha = 1$  we recover (2), but when  $\alpha < 1$  we obtain stronger inclusions. In Main Theorem 1, we use generic slowdowns in the alternation-trading proof from [25] and characterize the lower bound we obtain as a function of the parameter  $\alpha$  in our generic slowdown rule. While the core idea of this proof is essentially the same as the presentation of the proof in [27] (and the result can be thought of as “putting  $\alpha$  everywhere”), our proof technique is somewhat different. In the full version of the paper, this different approach yields a shorter proof of optimality than the one presented by Buss and Williams [7].

### 1.2.3 A generic slowdown rule from Grover search

In order to apply Main Theorem 1 to  $\mathcal{C} = \text{QCMATIME}[n]$  and obtain a better lower bound for QCMA, we show that the assumption  $\text{QCMATIME}[n] \subseteq \text{TS}[n^c]$  implies a generic slowdown rule for  $\alpha = \frac{2}{3}$ . Recall that Grover’s algorithm lets us search a space of size  $N$  with only  $O(\sqrt{N})$  quantum queries. We obtain our stronger slowdown rule by showing that Grover’s algorithm can be used to more efficiently remove the  $(x \log n)$ -bit quantifiers that arise after applications of the speedup rule, such as (1). In the  $\text{NTIME}$  vs  $\text{TS}$  setting, there are two ways to remove an  $(x \log n)$ -bit quantifier. First, we could remove it with an  $O(n^x)$

multiplicative blowup, by having the verifier exhaustively search through all strings of length  $x \log n$ . However, naively running  $n^x$  trials of an  $n^{d-x}$  computation would take  $n^d$  time, and our simulation would end up no faster than it was initially. Second, we may try to use a slowdown rule as in (2), but this incurs a runtime cost that depends on  $c$ . This option becomes expensive as we try to increase  $c$  and prove stronger lower bounds. Our key insight is that if our verifier is allowed to be quantum, Grover’s algorithm can be applied to perform this quantifier elimination in only  $O(n^{x/2})$  extra time overhead, independent of  $c$ . Then, by applying the assumption  $\text{QCMATIME}[n] \subseteq \text{TS}[n^c]$ , we can remove this quantum verifier along with the quantifier  $(\exists n^x)$  and ultimately demonstrate the inclusions

$$\begin{aligned}
& \dots (Qn^a)(\neg Qn^b)\text{TS}[n^d] \\
& \subseteq \dots (Qn^a)(\neg Qn^b)(Qn^x)(\neg Qx \log n)\text{TS}[n^{d-x}] && \text{Speedup Rule} \\
& \subseteq \dots (Qn^a)(\neg Qn^b)(Qn^x)\text{BQTIME}[n^{d-\frac{x}{2}}] && \text{Grover's Algorithm} \\
& \subseteq \dots (Qn^a)(\neg Qn^b)\text{TS}[n^{c \cdot \max(b, x, d-\frac{x}{2})}]. && \text{Assumption on QCMATIME}[n]
\end{aligned}$$

Letting  $x := \frac{2d}{3}$ , we obtain a generic slowdown rule with  $\alpha = \frac{2}{3}$ .

### 1.2.4 Lower bounds against randomized logspace

Some obstacles arise when trying to prove Main Theorem 3, which shows lower bounds against BPTS. The main problem is that the usual speedup rules for deterministic computation do not tell us how to use quantifiers to speedup randomized small-space computations. Fortunately, this particular issue was resolved by Diehl and Van Melkebeek [10], who gave a speedup rule for small-space randomized machines by coupling Nisan’s space-bounded derandomization [20] with the Sipser-Gács-Lautemann theorem [16]. This speedup rule, while somewhat less efficient than the speedup rule for deterministic machines, is still enough to obtain interesting superlinear time lower bounds. Applying this speedup rule, similar arguments as used in Main Theorem 1 yield the desired lower bounds.

## 1.3 Future Work

As mentioned earlier, the main advantage of the generic slowdown framework and Main Theorem 1 is that improvements in slowdown rules translate immediately into stronger bounds against TS. To this end, we highlight two particularly interesting directions.

1. Is it possible to prove a “quantum speedup rule”, whereby the runtime of quantum computations could be reduced by adding quantifiers (possibly over quantum states)? Presently, we are forced to use a slowdown rule to remove a quantum verifier from an alternating complexity class as soon as it is added. Having a quantum speedup rule would enable us to work with the quantum verifier before removing it, drastically widening the scope for potential alternation-trading proofs. It’s not hard to show that even certain weak forms of a quantum speedup rule would improve the generic slowdown parameter  $\alpha$  we can obtain in the QCMA vs. TS setting. Speedup rules also have applications outside complexity theory. For example, versions of speedup rules for low-space computation appear in the construction of delegation schemes in cryptography [4, 13, 14], and quantum speedup rules could play a part in extending such work to the quantum domain.

2. Can we use the ideas of this paper to improve existing lower bounds on counting-type ( $\#P$  related) problems, such as  $\#SAT$  and  $MAJ-SAT$ ? For example, could our super-quadratic time lower bound for  $QCMATIME[n]$  be somehow applied to obtain super-quadratic lower bounds for  $\#SAT$  as well? Because

$$MATIME[t] \subseteq QCMATIME[t] \subseteq PPTIME[t^{2+o(1)}]$$

as discussed in Section 1.1.2, lower bounds on  $QCMATIME[n]$  and  $MATIME[n]$  do translate to some lower bounds for counting problems against small-space. However, the known reductions from classes like  $MA$  and  $QCMA$  to counting problems incur a quadratic blowup. Furthermore, there is evidence that a quadratic blowup is necessary for black-box techniques [8, 9]. As such, it appears we must either improve the lower bound exponent, or prove that we can bypass the quadratic blowup outside of black-box settings.

## 1.4 Organization

Section 2 covers relevant background, especially on alternation-trading proofs. In Section 3, we study alternation-trading proofs with generic slowdowns and prove Main Theorem 1. In Section 4, we use Grover's algorithm to prove Lemma 30, allowing us to obtain a generic slowdown with  $\alpha = \frac{2}{3}$  and prove Main Theorem 2. In Section 5, we prove Main Theorem 3.

## 2 Preliminaries

We assume familiarity with classical complexity and quantum computing on the levels of [2] and [19], respectively.

### 2.1 Alternation-Trading Proofs

We start by defining various time-space complexity classes particular to this work. For a simple example of an alternation-trading proof, see Section 2.1.1 of the full version.

► **Definition 1.**  $TS[t(n)]$  is the class of languages computable by a deterministic random-access machine using space  $n^{o(1)}$  and time  $\tilde{O}(t(n))$  on an  $n$ -bit input.  $BPTS[t(n)]$  is the class of languages computable by a two-sided error randomized random-access machine using space  $O(\log n)$  and time  $\tilde{O}(t(n))$  on an  $n$ -bit input.

Note that a *randomized* random-access machine with random access to its input has only read-once access to its randomness.

► **Definition 2.** For positive constants  $\{a_i\}_{i \geq 1}$  and  $\{b_i\}_{i \geq 1}$  and quantifiers  $Q_i \in \{\exists, \forall\}$ , the **alternating complexity class**  $(Q_1 n^{a_1})^{b_1} (Q_2 n^{a_2})^{b_2} \dots (Q_k n^{a_k})^{b_k} TS[n^d]$  is the set of languages decidable by a machine operating in the following fashion on an  $n$ -bit input. Computation occurs in  $k + 1$  stages. In the  $i^{th}$  stage, for  $1 \leq i \leq k$ , the machine obtains from  $Q_i$  a string of length  $\tilde{O}(n^{a_i})$ . It then uses an  $n^{o(1)}$ -space machine and  $\tilde{O}(n^{b_i})$  time to compute  $\tilde{O}(n^{b_i})$  bits that are passed on to the next stage, taking as input the  $\tilde{O}(n^{a_i})$  bit string from the quantifier and the  $\tilde{O}(n^{b_{i-1}})$ -bit input from the previous stage of computation. The input to the first stage ( $i = 1$ ) is the  $n$ -bit input string itself. The verifier at the end receives an  $\tilde{O}(n^{b_k})$ -bit input and uses a  $n^{o(1)}$ -space machine and  $\tilde{O}(n^d)$  time to compute the final answer. The criteria for acceptance and rejection are analogous to those for  $\Sigma_k P$  and  $\Pi_k P$ .

Note that our notation obscures  $n^{o(1)}$  factors everywhere, although we may occasionally write out small factors for clarity. We index our  $b_i$  differently from the notation of [27], as our  $b_i$  is their  $b_{i+1}$ . This difference will be immaterial.

► **Definition 3.** Given an alternating complexity class  $(Q_1 n^{a_1})^{b_1} \dots (Q_k n^{a_k})^{b_k} \text{TS}[n^d]$ , we will refer to  $n^d$  as the **verifier runtime**.

► **Lemma 4** (Speedup Lemma [18][15]). For every  $0 < x < d$ ,

$$\text{TS}[n^d] \subseteq (Q n^x)^{\max(x,1)} (\neg Q x \log n)^1 \text{TS}[n^{d-x}].$$

Because our notation obscures  $n^{o(1)}$  factors, we may write this as

$$\text{TS}[n^d] \subseteq (Q n^x)^{\max(x,1)} (\neg Q n^0)^1 \text{TS}[n^{d-x}]. \quad (3)$$

**Proof.** We will prove the lemma when  $Q = \exists$ ; the other case follows because  $\text{TS}[n^d]$  is closed under complementation. The idea is that we can break up the transcript of a deterministic computation of length  $n^d$  into  $n^x$  pieces each of length  $n^{d-x}$ . Let  $M$  be an  $n^d$  time machine using  $n^{o(1)}$  space. On an  $n$ -bit input, our  $\Sigma_2$  machine will:

1. **Existentially** guess  $n^x - 1$  intermediate machine configurations  $X_1, \dots, X_{n^x-1}$  of  $M$ , each of size  $n^{o(1)}$ . These are passed, along with the input, to the next stage. This corresponds to the  $(Q n^x)^{\max(x,1)}$  part of the class in (3).
2. **Universally** quantify over all intermediate configurations, picking one. There are  $n^x$  pieces so our quantifier only needs  $O(\log n^x) \leq \tilde{O}(n^0)$  bits. If the quantifier picks the  $i^{\text{th}}$  configuration, then we pass the state pair  $(X_{i-1}, X_i)$  (along with the input) on to the next stage. We take  $X_0$  to be the initial configuration of  $M$ , and  $X_n$  to be the (WLOG unique) accepting machine configuration. This corresponds to the  $(\neg Q n^0)^1$  part in (3).
3. Given input  $x$  and a pair of configurations  $(X, Y)$  of  $M$ , the verifier simulates  $M$  starting at  $X$  for  $n^{d-x}$  steps, accepting if the configuration at the end is  $Y$  and rejecting otherwise. This corresponds to the  $\text{TS}[n^{d-x}]$  part of (3).

This completes the proof. ◀

As an extension, we may derive the speedup rule that we will use throughout this paper.

► **Corollary 5** (“Usual” Speedup Rule, [27]). For every  $0 < x < d$ ,

$$\begin{aligned} & (Q_1 n^{a_1})^{b_1} \dots (Q_k n^{a_k})^{b_k} \text{TS}[n^d] \\ & \subseteq (Q_1 n^{a_1})^{b_1} \dots (Q_k n^{\max(a_k, x)})^{\max(b_k, x)} (Q_{k+1} x \log n)^{b_k} \text{TS}[n^{d-x}]. \end{aligned}$$

**Proof.** Observe that we may merge together two quantifiers of the same type. Thus, taking  $Q = Q_k$  in Lemma 4, we find that

$$\begin{aligned} & (Q_1 n^{a_1})^{b_1} \dots (Q_k n^{a_k})^{b_k} \text{TS}[n^d] \\ & \subseteq (Q_1 n^{a_1})^{b_1} \dots (Q_k n^{a_k})^{b_k} (Q_k n^x)^{\max(b_k, x)} (Q_{k+1} x \log n)^{b_k} \text{TS}[n^{d-x}] \\ & \subseteq (Q_1 n^{a_1})^{b_1} \dots (Q_k n^{\max(a_k, x)})^{\max(b_k, x)} (Q_{k+1} x \log n)^{b_k} \text{TS}[n^{d-x}] \end{aligned}$$

where the second containment follows from Lemma 4. ◀

One might wonder whether we can do any better by also considering the containment arising from taking  $Q = \neg Q_k$  in Lemma 4. It turns out that any alternation-trading proof using this latter rule can be obtained with Corollary 5, and therefore we may safely ignore this option. This is Lemma 3.2 of [27].

► **Definition 6.** We refer to the value  $x$  in an application of the speedup rule as the **speedup parameter** for that application.



In Section 5, we will work with alternating complexity classes with randomized space-bounded verifiers, rather than deterministic ones. We will use a speedup rule due to Diehl and van Melkebeek [10].

► **Theorem 7** ([10]).  $\text{BPTS}[n^d] \subseteq (\forall n^0)^1 (\exists n^x)^{\max(x,1)} (\forall x \log n) \text{TS}[n^{d-x}]$ .

A sketch of the proof is in the full version. Note again that our notation allows us to hide  $n^{o(1)}$  factors. We chose not to obscure the last  $x \log n$  bits, as they will be extremely relevant later when we use Grover’s algorithm to remove  $O(\log n)$ -bit quantifiers.

As before, we may express Theorem 7 as a rule that can be applied to alternating complexity classes.

► **Corollary 8** (The “Randomized” Speedup Rule). *For every  $0 < x < d$ ,*

$$\begin{aligned} & (Q_1 n^{a_1})^{b_1} \dots (Q_k n^{a_k})^{b_k} \text{BPTS}[n^d] \\ & \subseteq (Q_1 n^{a_1})^{b_1} \dots (Q_k n^{a_k})^{b_k} (Q_{k+1} n^x)^{\max(b_k, x)} (Q_{k+2} x \log n)^{b_k} \text{TS}[n^{d-x}]. \end{aligned}$$

Note that unlike Corollary 5, which adds two quantifiers to speed up a deterministic computation, Corollary 8 adds three<sup>7</sup>.

We’ve already introduced the usual slowdown lemma, which we restate for convenience.

► **Lemma 9** (Slowdown Lemma [11]). *Assume that  $\text{NTIME}[n] \subseteq \text{TS}[n^c]$  for some  $c > 1$ . Then for all  $d \geq 1$ ,  $\text{NTIME}[n^d] \cup \text{coNTIME}[n^d] \subseteq \text{TS}[n^{cd}]$ .*

The slowdown rule follows from a padding argument and the observation that  $\text{TS}[n^c]$  is closed under complementation.

► **Corollary 10** (“Usual” Slowdown Rule [27]). *Assuming  $\text{NTIME}[n] \subseteq \text{TS}[n^c]$ , we have*

$$\begin{aligned} & (Q_1 n^{a_1})^{b_1} \dots (Q_k n^{a_k})^{b_k} \text{TS}[n^d] \\ & \subseteq (Q_1 n^{a_1})^{b_1} \dots (Q_{k-1} n^{a_{k-1}})^{b_{k-1}} \text{TS}[n^{c \cdot \max(d, b_k, a_k, b_{k-1})}]. \end{aligned}$$

Note that the exponent  $b_{k-1}$  is present in the maximum, because our assumption is  $\text{NTIME}[n] \subseteq \text{TS}[n^c]$  rather than  $\text{NTIME}[n^\delta] \subseteq \text{TS}[n^{c\delta}]$  for all  $\delta > 0$ .

► **Definition 11.** *Let  $c, \alpha \in \mathbb{R}$  such that  $0 < \alpha \leq 1 < c$ . A **generic slowdown rule with parameters  $\alpha$  and  $c$**  shows that*

$$(Q_1 n^{a_1})^{b_1} \dots (Q_k n^{a_k})^{b_k} \text{TS}[n^d] \subseteq (Q_1 n^{a_1})^{b_1} \dots (Q_{k-1} n^{a_{k-1}})^{b_{k-1}} \text{TS}[n^{c \cdot \max(\alpha d, b_k, a_k, b_{k-1})}].$$

Intuitively, having a generic slowdown rule with parameters  $c$  and  $\alpha$  means that we can turn classes like

$$\exists \forall \dots \exists \forall \text{TIME}[n^d]$$

into

$$\exists \forall \dots \forall \text{TIME}[n^{\alpha c d}].$$

We are now ready to define alternation-trading proofs.

<sup>7</sup> In both cases, the first quantifier is absorbed into the previous quantifier if one exists, in which case the number of “new” quantifiers is one and two respectively.

► **Definition 12** ([27]). An **alternation-trading proof** is a list of alternating complexity classes, where each subsequent class on the list is contained in the previous class. Each class is derived from the previous by applying one of the following rules.

1. If the class is  $\text{TS}[n^d]$  (i.e., the verifier is deterministic and there are no quantifiers), we may apply Lemma 4:

$$\text{TS}[n^d] \subseteq (\exists n^x)^{\max(x,1)} (\forall x \log n)^1 \text{TS}[n^{d-x}]$$

for some  $x \in (0, d)$ .

2. If the class has at least one quantifier and the verifier is deterministic (i.e., the class ends with  $\text{TS}[n^d]$ ), we may apply Corollary 5:

$$\begin{aligned} & (Q_1 n^{a_1})^{b_1} \dots (Q_k n^{a_k})^{b_k} \text{TS}[n^d] \\ & \subseteq (Q_1 n^{a_1})^{b_1} \dots (Q_k n^{\max(a_k, x)})^{\max(b_k, x)} (Q_{k+1} x \log n)^{b_k} \text{TS}[n^{d-x}] \end{aligned}$$

for some  $x \in (0, d)$ .

3. If the class is  $\text{BPTS}[n^d]$  (i.e., the verifier is randomized and there are no quantifiers), we may apply Theorem 7:

$$\text{BPTS}[n^d] \subseteq (\exists n^0)^1 (\forall n^x)^{\max(x,1)} (\exists x \log n) \text{TS}[n^{d-x}]$$

for some  $x \in (0, d)$ .

4. If the class has at least one quantifier and the verifier is randomized (i.e., the class ends with  $\text{BPTS}[n^d]$ ), we may apply Corollary 8:

$$\begin{aligned} & (Q_1 n^{a_1})^{b_1} \dots (Q_k n^{a_k})^{b_k} \text{BPTS}[n^d] \\ & \subseteq (Q_1 n^{a_1})^{b_1} \dots (Q_k n^{a_k})^{b_k} (Q_{k+1} n^x)^{\max(b_k, x)} (Q_{k+2} x \log n)^{b_k} \text{TS}[n^{d-x}] \end{aligned}$$

for some  $x \in (0, d)$ .

5. If the class has at least one quantifier, and a generic slowdown rule with parameters  $\alpha$  and  $c$  hold for the class, we may apply it:

$$\dots (Q_k n^{a_k})^{b_k} (\text{BP}) \text{TS}[n^d] \subseteq \dots (Q_{k-1} n^{a_{k-1}})^{b_{k-1}} (\text{BP}) \text{TS}[n^{c \cdot \max(\alpha d, b_k, a_k, b_{k-1})}].$$

We say that an alternation-trading proof **shows a contradiction for  $c$**  if it contains an application of a speedup rule and the proof shows either  $\text{TS}[n^d] \subseteq \text{TS}[n^{d'}]$  for  $d' \leq d$  or  $\text{BPTS}[n^d] \subseteq \text{BPTS}[n^{d'}]$  for  $d' \leq d$ .

Note that rules 3 and 4 only apply when proving lower bounds against BPTS.

The containment  $\text{TS}[n^d] \subseteq \text{TS}[n^{d'}]$  for  $d \geq d'$  does not automatically yield a contradiction<sup>8</sup>. Fortunately however, we are still able to derive contradictions from this.

► **Theorem 13** (Lemma 3.1 of [27]). If, under the assumption  $\text{NTIME}[n] \subseteq \text{TS}[n^c]$ , there is an alternation-trading proof with at least two inclusions showing that  $\text{TS}[n^d] \subseteq \text{TS}[n^{d'}]$  for  $d' \leq d$ , then the assumption must have been false and  $\text{NTIME}[n] \not\subseteq \text{TS}[n^c]$

<sup>8</sup> To apply the naive approach, we need a single machine in  $\text{TS}[n^d]$  that can simulate everything in  $\text{TS}[n^{d'}]$ . However, any fixed machine in  $\text{TS}[n^d]$  cannot simulate things use more space than it does. If our simulating machine in  $\text{TS}[n^d]$  uses space  $f(n) = n^{o(1)}$  then there is always a machine in  $\text{TS}[n^{d'}]$  that uses more space while still being  $n^{o(1)}$  and our simulating machine cannot simulate this one.

In Section 5, we will show that similar statements hold for BPTS in the contexts in which we need them to hold, thus allowing us to derive contradictions from  $\text{BPTS}[n^d] \subseteq \text{BPTS}[n^{d'}]$  for  $d' \leq d$ .

► **Definition 14** ([27]). *An alternating complexity class  $(Q_1 n^{a_1})^{b_1} \dots (Q_k n^{a_k})^{b_k} (\text{BP})\text{TS}[n^d]$  is **orderly** if for all  $i \in [k]$ ,  $a_i \leq b_i$ .*

► **Lemma 15** ([27]). *Any alternation-trading proof using rules from Definition 12 is orderly.*

As noted by Buss and Williams [7], Lemma 15 implies that, when describing an alternation-trading proof consisting of speedups and slowdowns, it is sufficient to only specify the  $\{b_i\}$  and disregard the  $\{a_i\}$ . We will be somewhat more careful when we apply Grover’s algorithm in the quantum setting, but when we can we will simplify our notation by writing only a single exponent inside the parenthesis. Thus, we may abuse notation to write alternating complexity classes in the form  $(Q_1 n^{a_1}) \dots (Q_k n^{a_k}) (\text{BP})\text{TS}[n^d]$ , where the  $a_i$  are then understood to be the maxima of the corresponding pairs of exponents in the full notation.

► **Definition 16** ([27]). *A **proof annotation** is a way of specifying a sequence of applications of speedup and slowdown rules. We write **1** to denote a speedup rule and **0** to denote a (possibly generic) slowdown rule. When appropriate, we put a subscript under a **1** to denote the speedup parameter used for that speedup rule application.*

In this paper, we will work with alternation-trading proofs which apply only one slowdown rule (many times). For such proofs, the sequence of speedups and slowdowns fully determines whether the verifier is deterministic or randomized at a given line of the proof. This means that, when specifying a proof annotation, we do not need to specify which speedup rule we are applying between Corollary 5 and Corollary 8. When the verifier is randomized we must apply Corollary 8, and when the verifier is deterministic we should always apply Corollary 5 as it is strictly more efficient than Corollary 8.

## 2.2 Computational Model

All functions used to bound runtime or space are assumed to be constructible in the given resources. Our model for classical computation is the space-bounded random-access machine, unless specified otherwise. Our proofs are robust to all notions of random access we know.

Our model for quantum computation will be that of Van Melkebeek and Watson [23], but our results hold for any reasonable quantum model capable of *obliviously* applying unitaries from a fixed universal set and with quantum random-access to the input.<sup>9</sup> Recall that if  $x \in \{0, 1\}^n$  is an input, an algorithm is said to have quantum random-access to  $x$  if it can perform the transformation

$$\sum_{i \in [n]} \alpha_i |i\rangle |b\rangle \mapsto \sum_{i \in [n]} \alpha_i |i\rangle |b \oplus x_i\rangle,$$

where  $i$  is an index and  $x_i$  denotes the bit at the  $i^{\text{th}}$  position of  $x$ . The model of [23] is capable of simulating all the usual models of quantum computing, and deals carefully with issues like intermediate measurements and numerical precision.

<sup>9</sup> Here, “obliviously” means that the unitaries applied depend only on the length of the input.

### 2.3 Some Atypical Complexity Classes

We stated our lower bounds in Section 1 in terms of QCMA and MA. However, our results actually hold for slightly smaller classes, which we describe below.

► **Definition 17.** *The complexity class  $\exists \cdot \text{BQP}$  is the set of languages  $L$  for which there exists a BQP verifier  $\mathcal{A}$  such that*

- $x \in L \implies (\exists w) \Pr[\mathcal{A}(x, w) = 1] \geq \frac{2}{3}$
- $x \notin L \implies (\forall w) \Pr[\mathcal{A}(x, w) = 1] \leq \frac{1}{3}$ .

*We will write  $\exists \cdot \text{BQP}_{s,c}$  to denote  $\exists \cdot \text{BQP}$  where Arthur has completeness  $c$  and soundness  $s$ . We will write  $\exists \cdot \text{BQTIME}[t(n)]$  to denote  $\exists \cdot \text{BQP}$  where the length of Merlin's proof and the runtime of the verifier are both  $O(t(n))$ .*

We may define  $\exists \cdot \text{BPP}$  and  $\exists \cdot \text{BPTIME}$  analogously.

► **Remark 18.** Note that, while  $\exists \cdot \text{BQP} \subseteq \text{QCMA} \subseteq \text{QMA}$  (respectively,  $\exists \cdot \text{BPP} \subseteq \text{MA}$ ), it is not clear if  $\exists \cdot \text{BQP} = \text{QCMA}$  ( $\exists \cdot \text{BPP} = \text{MA}$ ) due to differences in the promise conditions. In  $\exists \cdot \text{BQP}$ , we require that the verifier  $\mathcal{A}(x, w)$  lie in BQP, meaning that it satisfies the promise on every input pair  $(x, w)$ . On the other hand, in the “yes” case of QCMA (when a string  $x$  is in the language), we require only that there exists a polynomial-sized witness  $y$  making the verifier  $\mathcal{A}(x, w)$  accept with probability exceeding  $\frac{2}{3}$ . This does not preclude the existence of a witness string  $w'$  such that  $\frac{1}{3} < \Pr[\mathcal{A}(x, w') = 1] < \frac{2}{3}$ .

## 3 Lower Bounds With Generic Slowdowns

We will start by introducing a method to reduce the verifier runtime of a class without increasing any of the quantifier exponents, assuming that the verifier runtime isn't too large. This was the main feature in the alternation-trading proofs of [26] that allowed improvement beyond the results of Lipton-Fortnow-Van Melkebeek-Viglas [11].

► **Lemma 19** (Generalizes [26]). *Let  $0 < \alpha \leq 1$  be a real number. Let  $c$  be a positive real such that  $c < \frac{1+\alpha}{\alpha}$ . Given any class*

$$\dots (Q_k n^{a_k}) \text{TS}[n^{a_{k+1}}]$$

*where  $ca_k \leq a_{k+1} < \frac{\alpha c}{\alpha c - 1} a_k$ , there is a nonnegative integer  $N := N(a_k)$  such that the annotation  $(10)^N \mathbf{0}$  with the appropriate speedup parameters proves that*

$$\dots (Q_k n^{a_k}) \text{TS}[n^{a_{k+1}}] \subseteq \dots (Q_k n^{a_k}) \text{TS}[n^{ca_k}] \subseteq \dots \text{TS}[n^{c^2 a_k}].$$

We prove this lemma in the full version. The use of  $a_k$  as the speedup parameter in Lemma 19 may seem arbitrary, but it will turn out that this is in fact the best speedup parameter in this setting. Based on this lemma, we can define a new rule that we may use in alternation-trading proofs.

► **Definition 20.** *Consider an alternating complexity class  $(Q_1 n^{a_1}) \dots (Q_k n^{a_k}) \text{TS}[n^d]$ . Given  $0 < \alpha \leq 1$  and  $c > 0$  satisfying  $c < \frac{1+\alpha}{\alpha}$ , we define the **wiggle rule** with parameters  $\alpha, c$  to be the following:*

- *If  $d < \frac{\alpha c}{\alpha c - 1} a_k$ , apply  $(1_{a_k} \mathbf{0})^t$  for  $t := \left\lceil \frac{a_{k+1}}{a_t} \right\rceil$ .*
- *Otherwise, do nothing.*

*We call an application of the wiggle rule **proper** if we are in the first case and **improper** otherwise. In a proof annotation, when  $\alpha$  and  $c$  are fixed, we will denote an application of the wiggle rule by 2.*

It is worth remembering the ratio

$$\frac{\alpha c}{\alpha c - 1}$$

as it will show up frequently in the remainder of this paper. Following the notation of Definition 20, if the value of  $d$  (the verifier runtime exponent) for an alternating complexity class is at most  $\frac{\alpha c}{\alpha c - 1}$  times  $a_k$  (the exponent in the final quantifier), we may reduce the verifier runtime to its smallest possible value  $ca_k$  with an application of Definition 20 and Lemma 19, as the following corollary shows.

► **Corollary 21** (Corollary of Lemma 19). *Consider a class*

$$(Q_1 n^{a_1}) \dots (Q_k n^{a_k}) \text{TS}[n^d].$$

*Given  $0 < \alpha \leq 1$  and  $c > 0$  satisfying  $c < \frac{1+\alpha}{\alpha}$ , applying the wiggle rule (Definition 20) yields the class:*

- $(Q_1 n^{a_1}) \dots (Q_k n^{a_k}) \text{TS}[n^{ca_k}]$ , if  $d < \frac{\alpha c}{\alpha c - 1} a_k$ .
- $(Q_1 n^{a_1}) \dots (Q_k n^{a_k}) \text{TS}[n^d]$  otherwise.

We are now in a position to prove Main Theorem 1, which we restate for convenience.

► **Main Theorem 1.** *Fix  $\alpha \in (0, 1]$  and let  $\mathcal{C}$  be a complexity class. Let  $r_1$  be the largest root of the polynomial  $P_\alpha(x) := \alpha^2 x^3 - \alpha x^2 - 2\alpha x + 1$ . If  $\mathcal{C} \subseteq \text{TS}[n^c]$  implies a generic slowdown rule with parameters  $\alpha$  and  $c$ , then  $\mathcal{C} \not\subseteq \text{TS}[n^c]$  for  $c < r_1$ .*

We will use the following lemma in the proof of Main Theorem 1; its proof can be found in the full version.

► **Lemma 22.** *If  $r_1$  and  $r_2$  are the two largest roots of  $P_\alpha(x) := \alpha^2 x^3 - \alpha x^2 - 2\alpha x + 1$  then*

$$r_2(\alpha) < \frac{1 + \sqrt{1 + 4\alpha}}{2\alpha} < r_1(\alpha) < \frac{1 + \alpha}{\alpha}$$

*for all  $\alpha > 0$ .*

**Proof of Main Theorem 1.** The lower bound will follow from applying the annotation  $1^k \mathbf{0}(20)^k$  as  $k \rightarrow \infty$ . (Recall that  $\mathbf{2}$  denotes an application of the wiggle rule of Definition 20.) We will choose speedup parameters  $\{x_i\}$  so that the following sequence of inclusions is valid and every application of the wiggle rule is proper.

$$\begin{aligned} \text{TS}[n^d] &\subseteq (\exists n^{x_1})(\forall n^{x_2}) \dots (Q_k n^{x_k})(Q_{k+1} n^{x_k}) \text{TS}[n^{(d-x_1-x_2-\dots-x_k)}] & 1^k \mathbf{0}(20)^k \\ &\subseteq (\exists n^{x_1})(\forall n^{x_2}) \dots (Q_k n^{x_k}) \text{TS}[n^{\alpha c(d-x_1-x_2-\dots-x_k)}] & 1^k \mathbf{0}(20)^k \\ &\subseteq (\exists n^{x_1})(\forall n^{x_2}) \dots (Q_k n^{x_k}) \text{TS}[n^{cx_k}] & 1^k \mathbf{0}20(20)^{k-1} \\ &\subseteq (\exists n^{x_1})(\forall n^{x_2}) \dots (Q_{k-1} n^{x_{k-1}}) \text{TS}[n^{\alpha c^2 x_k}] & 1^k \mathbf{0}20(20)^{k-1} \\ &\subseteq (\exists n^{x_1})(\forall n^{x_2}) \dots (Q_{k-1} n^{x_{k-1}}) \text{TS}[n^{cx_{k-1}}] & 1^k \mathbf{0}2020(20)^{k-2} \\ &\dots & 1^k \mathbf{0}2020(20)^{k-2} \\ &\subseteq \text{TS}[n^{\alpha c^2 x_1}]. \end{aligned}$$

In order to ensure a contradiction at the end, we will set  $x_1 := \frac{d}{\alpha c^2}$ . In order to ensure that the first application of the wiggle rule is proper, we require that the  $\{x_i\}$  satisfy

$$\alpha c(d - x_1 - \dots - x_k) < \frac{\alpha c}{\alpha c - 1} x_k \iff d - x_1 - x_2 - \dots - x_k < \frac{1}{\alpha c - 1} \cdot x_k. \quad (4)$$

In order to ensure that we can apply the wiggle rule properly in all other steps, we require that the  $\{x_i\}$  satisfy

$$\alpha c^2 x_i < \frac{\alpha c}{\alpha c - 1} \cdot x_{i-1} \quad (5)$$

for all  $2 \leq i \leq k$ .

The next claim lets us find valid speedup parameters  $\{x_i\}$  for certain values of  $c$  depending on the number of iterations  $k$  that we allow in our annotation. The expression will look somewhat hairy, but fortunately most of it will disappear in the limit as  $k$  becomes large.

▷ **Claim 23.** Pick  $\epsilon > 0$ , and let  $\tau := \frac{1-\epsilon}{c(\alpha c-1)}$ . There are choices of the speedup parameter  $x_i$  such that the annotation  $\mathbf{1}^k \mathbf{0}(\mathbf{20})^k$  implies a contradiction for all  $c$  satisfying

$$\alpha c^2 - \frac{\tau^k - 1}{\tau - 1} < \frac{\tau^k}{\alpha c - 1}. \quad (6)$$

Proof of Claim. Let  $x_i := \left(\frac{1-\epsilon}{c(\alpha c-1)}\right)^{i-1} x_1 = \left(\frac{1-\epsilon}{c(\alpha c-1)}\right)^{i-1} \frac{d}{c^2}$  for  $i \geq 2$ . Observe that all the  $x_i$  are positive and satisfy the constraints of (5) for all  $i$ . If we take  $d$  to be sufficiently large we can ensure that every exponent in the proof exceeds 1. Therefore, if we can show that our selection of  $\{x_i\}$  satisfies the first constraint, (4), we will have a valid sequence of rules and, by our choice of  $x_1$  above, have derived a contradiction.

Plugging our choice of  $\{x_i\}$  into (4) yields the constraint

$$d - \frac{d}{\alpha c^2} \left(1 + \frac{1-\epsilon}{c(\alpha c)} + \cdots + \left(\frac{1-\epsilon}{c(\alpha c-1)}\right)^{k-1}\right) < \frac{1-\epsilon}{\alpha c-1} \left(\left(\frac{1-\epsilon}{c(\alpha c-1)}\right)^{k-1} \frac{d}{\alpha c^2}\right). \quad (7)$$

This is equivalent to

$$\alpha c^2 - \frac{\tau^k - 1}{\tau - 1} < \frac{\tau^k}{\alpha c - 1}, \quad (8)$$

as desired. ◁

The lemma condition  $c > \frac{1+\sqrt{1+4\alpha}}{2\alpha}$  means that  $(c(\alpha c-1))^{-1} < 1$ . Now, let  $\epsilon := \frac{1}{k}$  and allow  $k \rightarrow \infty$  in (6). Since  $\tau \rightarrow (c(\alpha c-1))^{-1}$  as  $k \rightarrow \infty$ , we see that (6) becomes

$$\alpha c^2 - \frac{c(\alpha c-1)}{c(\alpha c-1)-1} < 0 \iff \alpha^2 c^3 - \alpha c^2 - 2\alpha c + 1 < 0$$

in the limit. More formally, we have shown that for every  $c$  satisfying this constraint, if we take  $k$  to be large enough, then our choice of  $\{x_i\}$  satisfies (4). The leading coefficient is positive so we satisfy all the constraints (i.e. the alternation-trading proof shows a contradiction) when  $c$  lies between the largest and second largest roots of this cubic. Implicitly, we require  $c < \frac{1+\alpha}{\alpha}$  so that we can apply Lemma 19 and  $c > \frac{1+\sqrt{1+4\alpha}}{2\alpha}$  so that the undesired terms in (6) vanish. By Lemma 22, these are satisfied when  $\frac{1+\sqrt{1+4\alpha}}{2\alpha} < c < r_1$ . Furthermore, if  $\mathcal{C} \not\subseteq \text{TS}[n^{c'}]$  for some  $c'$  then we automatically have  $\mathcal{C} \not\subseteq \text{TS}[n^c]$  for all  $c < c'$ . ◀

From Main Theorem 1, we immediately obtain the following corollary.

► **Corollary 24** ([25]). For  $c < 2 \cos(\frac{\pi}{7}) \approx 1.801$ ,  $\text{NTIME}[n] \not\subseteq \text{TS}[n^c]$ .

**Proof.** Taking  $\alpha = 1$  (as this is normal slowdown) yields  $c^3 - c^2 - 2c + 1 < 0$ . The largest root is  $2 \cos(\frac{\pi}{7})$ , so we have  $\text{NTIME}[n] \not\subseteq \text{TS}[n^{2 \cos(\frac{\pi}{7}) - o(1)}]$  ◀

## 4 A Generic Slowdown Rule From Grover's Algorithm

Now that we've proved the general case, we turn to an application. Observe that any application of a speedup rule with parameter  $x$  results in an alternating complexity class whose final quantifier is over  $x \log n$  bits, as this quantifier indexes over the  $n^x$  states that it receives from the penultimate quantifier. For example,

$$\text{TS}[n^d] \subseteq (\exists n^x)(\forall x \log n)^1 \text{TS}[n^{d-x}].$$

Normally, we could remove the last quantifier with a slowdown rule, yielding the inclusion

$$\text{TS}[n^d] \subseteq (\exists n^x) \text{TS}[n^{c \cdot \max(x, d-x, 1)}]. \quad (9)$$

We could also remove the quantifier by having the deterministic verifier try all  $n^x$  possible strings it could receive, but this yields the useless inclusion  $\text{TS}[n^d] \subseteq (\exists n^x) \text{TS}[n^d]$ . However, if we allow alternating complexity classes with quantum verifiers rather than deterministic verifiers, we can remove the last quantifier more efficiently. In particular, we can think of the last quantifier as a search problem over a space of size  $N := n^x$ . Classically, no blackbox algorithm can search over  $N$  items with fewer than  $N$  queries in the worst case, but in the quantum setting Grover's algorithm lets us do this in  $O(\sqrt{N})$  queries! This gives us, for some informal notion of quantum time,

$$\text{TS}[n^d] \subseteq (\exists n^x) \text{QTIME}[n^{d-\frac{x}{2}}]. \quad (10)$$

This method of quantifier removal allows us to remove quantifiers more efficiently than (9) when  $c$  is large at the cost of introducing a quantum verifier. However, the quantum verifier can be replaced with a deterministic verifier, under the assumption  $\text{QCMATIME}[n] \subseteq \text{TS}[n^c]$ ! Ultimately, we will find that combining a speedup (adding one quantifier), (10) (removing one quantifier), and the assumption (removing one quantifier) yields a generic slowdown rule with  $\alpha = \frac{2}{3}$ .

### 4.1 Review of Grover's Algorithm

Given (quantum) query access to a function  $f: [N] \rightarrow \{0, 1\}$ , Grover's algorithm tells us whether or not there exists an  $\alpha \in [N]$  such that  $f(\alpha) = 1$ . For a given  $f$ , let  $S := \{\alpha \in [N]: f(\alpha) = 1\}$ . It turns out, per [12], that the probability of success of Grover search after  $j$  iterations is  $\sin^2(2(j+1)\theta)$  where  $\theta = \sin^{-1} \sqrt{\frac{|S|}{N}}$ . Regardless of  $\frac{|S|}{N}$ , a sufficiently large random number of iterations should succeed with probability roughly  $\frac{1}{2}$ , which is the average value of  $\sin^2 x$ . The following lemma of [5] formalizes this.

► **Lemma 25** ([5]). *Let  $k$  be an arbitrary positive integer and let  $j$  be an integer chosen uniformly at random from  $[0, k-1]$ . If we observe the register after applying  $j$  Grover iterations starting from the uniform state, the probability of obtaining a solution is at least  $\frac{1}{4}$  when  $k \geq \frac{1}{\sin 2\theta}$ .*

► **Corollary 26.** *Let  $j$  be an integer chosen uniformly at random from  $[0, (\sin \frac{2}{\sqrt{N}})^{-1}]$ . If we observe the register after applying  $j$  Grover iterations starting from the uniform state, the probability of obtaining a solution is at least  $\frac{1}{4}$ .*



## 4.2 Using Grover's Algorithm to Invert RAMs

In order to apply Grover's algorithm to remove the quantifier in expressions like  $(\exists \log n) \text{TS}[n^d]$  – specifically, to perform Grover diffusion – we must be able to implement the relevant function  $f \in \text{TS}[n^d]$  in our quantum computational model. The following lemma converts the classical random-access machine of  $f$  to a “normal form” that is more amenable to implementation in a quantum computer. The workspace of  $A$  includes an input address register used to specify which bit of the input the machine wishes to read and an input query bit which stores the result of the most recent query of the input. Assume without loss of generality that the first  $\lceil \log n \rceil$  bits of the workspace hold the input address register, and bit  $\lceil \log n \rceil + 1$  holds the current bit of input being read.

► **Lemma 27** (Normal Form Circuit for Time-Space Bounded Computation). *Let  $f$  be a function computed by a random-access machine  $A$  in time  $t(n)$  and space  $s(n)$  on inputs of length  $n$ . Then, there exists a sequence of uniform Boolean circuits  $\mathcal{C}_1, \dots, \mathcal{C}_r$ , such that:*

1. *for all  $1 < i \leq r - 1$ ,  $\mathcal{C}_i$  has  $s$  inputs and  $s$  outputs,*
2.  *$\mathcal{C}_1$  has no inputs and  $s$  outputs,*
3.  *$\mathcal{C}_r$  has  $s$  inputs and 1 output,*
4.  *$\sum_{i=1}^r |\mathcal{C}_i| = O(ts^2)$  where  $|\mathcal{C}_i|$  denotes the size of  $\mathcal{C}_i$ , and furthermore*

$$f(y) = \mathcal{C}_r \circ R \circ \mathcal{C}_{r-1} \circ \dots \circ R \circ \mathcal{C}_1, \quad (11)$$

where  $R: \{0, 1\}^{s(n)} \rightarrow \{0, 1\}^{s(n)}$ , on input  $z \in \{0, 1\}^{s(n)}$ , sets  $z_{\lceil \log n \rceil + 1} = y_{z_1 z_2 \dots z_{\lceil \log n \rceil}}$  and leaves the remainder of  $z$  unchanged. Here, we write  $s_j$  to denote the  $j^{\text{th}}$  bit of a string  $s$ .

In Lemma 27, the circuits  $\mathcal{C}_i$  are used to simulate steps in the workspace of the machine, and the function  $R$  perform random accesses to the  $n$ -bit input  $y$ , by setting its  $(\lceil \log n \rceil + 1)^{\text{th}}$  output bit to the bit of  $y$  whose index is specified by the first  $\lceil \log n \rceil$  bits. Note that  $y$  is the original length- $n$  input to  $f$  and  $n$  may be much larger than  $s(n)$ . Lemma 27, along with Corollary 28 allow us to translate a RAM to a quantum circuit with  $s(n)$  wires and with intermittent QRAM calls: the Boolean circuits  $\{\mathcal{C}_i\}$  can be replaced with quantum circuits in the usual way and each  $R$  can be replaced by a QRAM call. Proofs of Lemma 27 and Corollary 28 can be found in the full version.

► **Corollary 28.** *Let  $f: \{0, 1\}^m \rightarrow \{0, 1\}$  be a function computable in classical time  $t$  and space  $s$  by a random-access machine. Then, the transformation  $|y\rangle |0\rangle \mapsto |y\rangle |f(y)\rangle$  can be implemented by a quantum computer with quantum random-access to its input in time  $O(ts^2)$ .*

We now state the main lemma of this subsection.

► **Lemma 29** (Grover's Algorithm for Time-Space Bounded Computation). *Given a function  $f: \{0, 1\}^m \rightarrow \{0, 1\}$  that can be computed in classical time  $t$  and space  $s$  by a random-access machine, there is a quantum algorithm taking time  $O(2^{\frac{m}{2}}(ts^2 + m))$  that finds a value  $\alpha \in \{0, 1\}^m$  such that  $f(\alpha) = 1$ , assuming one exists, with probability at least  $\frac{2}{3}$ . In particular, when  $m = x \log n$ ,  $t = n^d$ ,  $s = O(\log n)$ , we obtain a time bound of  $\tilde{O}(n^{d+\frac{x}{2}})$ .*

**Proof.** (Sketch) We run Grover's algorithm. Per Corollary 26, we need  $O(2^{\frac{m}{2}})$  Grover iterations. Grover diffusion is dominated by the cost of implementing the function itself, which by Corollary 28 is  $O(ts^2)$ . Inversion about the mean takes  $O(m)$  time. ◀

### 4.3 Proving Main Theorem 2

We are now in a position to apply Grover's algorithm to prove a generic slowdown rule.

► **Lemma 30.** *If  $\exists \cdot \text{BQTIME}_{\frac{1}{3},1}[n] \subseteq \text{TS}[n^c]$  then*

$$\dots (Q_{k-1}n^{a_{k-1}})^{b_{k-1}} (Q_k n^{a_k})^{b_k} \text{TS}[n^d] \subseteq \dots (Q_{k-1}n^{a_{k-1}})^{b_{k-1}} \text{TS}[n^{c \cdot \max(a_k, b_k, b_{k-1}, 1, \frac{2d}{3})}].$$

Therefore, under the assumption  $\exists \cdot \text{BQTIME}_{\frac{1}{3},1}[n] \subseteq \text{TS}[n^c]$  we may apply a generic slowdown rule with parameters  $\alpha = \frac{2}{3}$  and  $c$ . Lemma 30 corresponds to the following sequence of operations:

1. Classical Speedup (Corollary 5)
2. Grover's algorithm to invert a classical function with classical input (Lemma 29)
3. Direct application of  $\exists \cdot \text{BQTIME}[n] \subseteq \text{TS}[n^c]$

**Proof.** By classical speedup, we have

$$\dots (Q_k n^{a_k})^{b_k} \text{TS}[n^d] \subseteq \dots (Q_k n^{\max(a_k, x)})^{\max(b_k, x)} (Q_{k+1} x \log n)^{b_k} \text{TS}[n^{d-x}]. \quad (12)$$

Consider the function  $g: \{0, 1\}^{n^{\max(b_k, x)}} \times \{0, 1\}^{x \log n} \rightarrow \{0, 1\}$  which implements the  $\text{TS}[n^{d-x}]$  verifier from the right-hand-side of (12) given as inputs the  $n^{\max(b_k, x)}$  bits of output from the  $k^{\text{th}}$  stage/quantifier and the  $(x \log n)$ -bit string chosen by the  $(k+1)^{\text{th}}$  stage/quantifier. We will apply Lemma 29 to the function  $g_z := g(z, \cdot)$ , where  $z$  is the output from the  $k^{\text{th}}$  stage of the class (not the  $(k+1)^{\text{th}}$  stage!). In particular, we can use Grover's algorithm to search over the space of possible values of the last quantifier of  $(x \log n)$  bits. Thus, the inputs to  $g_z$  are strings of length  $m = x \log n$ . The runtime of  $g_z$  is  $O(ts^2)$  by Corollary 28, as  $g_z$  just needs to evaluate the verifier on  $z$  (the output of the  $k^{\text{th}}$  stage) and a length  $m$ -input. Therefore, applying Lemma 29,

$$\subseteq \dots (Q_{k-1}n^{a_{k-1}})^{b_{k-1}} (Q_k n^{\max(a_k, x)})^{\max(b_k, x)} \text{BQTIME}[n^{d-\frac{x}{2}}].$$

Without loss of generality, suppose that  $Q_k = \exists$ . Then, we can continue the sequence of inclusions as follows:

$$\begin{aligned} &\subseteq \dots (Q_{k-1}n^{a_{k-1}})^{b_{k-1}} \exists \cdot \text{BQTIME}[n^{\max(a_k, b_k, x, d-\frac{x}{2})}] \\ &\subseteq \dots (Q_{k-1}n^{a_{k-1}})^{b_{k-1}} \text{TS}[n^{c \cdot \max(b_{k-1}, a_k, b_k, x, d-\frac{x}{2}, 1)}]. \end{aligned}$$

(Note we need a 1 in the maximum in the last equation, because our assumption  $\exists \cdot \text{BQTIME}_{\frac{1}{3},1}[n] \subseteq \text{TS}[n^c]$  only implies  $\exists \cdot \text{BQTIME}[n^\delta] \subseteq \text{TS}[n^c]$  for  $\delta < 1$ .) To minimize the exponent, we take  $x = \frac{2a_0}{3}$ , yielding the exponent in the lemma statement. ◀

► **Remark 31.** Note that in the proof of Lemma 30 we do not need to account for the  $n^{b_k}$  exponent on the  $(k+1)^{\text{th}}$  stage/quantifier when computing the runtime of  $g_z$  when preparing to apply Lemma 29. This is because the quantifier  $(Q_{k+1} x \log n)^{b_k}$  on the right-hand-side of (12), which arises due to a speedup rule, has a  $b_k$  in the exponent only because it needs to copy and pass on the output of the  $k^{\text{th}}$  stage. This is normally necessary because the verifier on the right-hand-side of (12) needs to run from configuration to configuration on the original verifier's input and hence needs to be able to access the original input (i.e., the input to the verifier on the left-hand-side of the inclusion). However, by our definition of  $g_z$ , we don't need to worry about the cost of copying the output from the  $k^{\text{th}}$  stage  $z$  is itself the output of the  $k^{\text{th}}$  stage. We don't need to copy and pass on the original input because  $g_z$  already has it! Put differently, because we're collapsing two stages into one, we don't need to expend time on computation that's only used to pass input along.

Note that every possible proof involving slowdown and Grover's algorithm (Lemma 29) can be carried out using Lemma 30 as the only rule for removing quantifiers, as we can only apply Lemma 29 between a speedup and a slowdown. (More details on why are in the full version.) Plugging  $\alpha = \frac{2}{3}$  into Main Theorem 1, we obtain the following corollary.

► **Corollary 32.**  $\exists \cdot \text{BQTIME}_{\frac{1}{3},1}[n] \not\subseteq \text{TS}[n^c]$  for  $c < \frac{3+\sqrt{3}}{2} \approx 2.366$ .

Since  $\exists \cdot \text{BQTIME}[n] \subseteq \text{QCMATIME}[n]$ , this proves Main Theorem 2.

## 5 Lower Bounds Against BPTS

We turn to proving lower bounds against randomized algorithms. In this section, we will use both Corollary 5 and Corollary 8 as speedup rules. The randomized slowdown rule is straightforward.

► **Lemma 33** (“Randomized” Slowdown Rule). *Assuming  $\exists \cdot \text{BPTIME}[n] \subseteq \text{BPTS}[n^c]$ , we have*

$$(Q_1 n^{a_1})^{b_1} \dots (Q_k n^{a_k})^{b_k} \text{BPTS}[n^d] \subseteq (Q_1 n^{a_1})^{b_1} \dots (Q_{k-1} n^{a_{k-1}})^{b_{k-1}} \text{BPTS}[n^{c \cdot \max(d, b_k, a_k, b_{k-1})}].$$

► **Lemma 34.** *Suppose that, under the assumption  $\exists \cdot \text{BPTIME}[n] \subseteq \text{BPTS}[n^c]$  for some  $c$ , there is an alternation-trading proof with at least two inclusions proving that  $\text{BPTS}[n^d] \subseteq \text{BPTS}[n^{d'}]$  for  $d < d'$ . Then, the assumption is false.*

A proof of this lemma can be found in the full version. Recall that by Lemma 15, we may suppress some of the exponents when writing alternating complexity classes. We will do so throughout the remainder of this section.

Now that we have the preliminaries out of the way, let us prove some lower bounds. The first part of Main Theorem 3 is an immediate corollary of the following theorem.

► **Theorem 35.**  $\exists \cdot \text{BPTIME}[n] \not\subseteq \text{TS}[n^c]$  for  $c < r_1$ , where  $r_1 \approx 1.466$  is the largest root of the polynomial  $x^3 - x^2 - 1 = 0$ .

**Proof.** Our analysis is similar to the proof of Main Theorem 1. The bound will arise by applying the annotation  $\mathbf{1}^k \mathbf{0}^{k+2}$  with the appropriate speedup parameters as  $k \rightarrow \infty$ . We will choose parameters  $\{x_i\}$  so that the following sequence of inclusions is valid.

$$\begin{aligned} \text{BPTS}[n^d] &\subseteq (\exists n^1)(\forall n^{x_1})(\exists n^{x_2}) \dots (Q_k n^{x_k})(Q_{k+1} n^{x_k}) \text{TS}[n^{(d-x_1-x_2-\dots-x_k)}] && \mathbf{1}^k \mathbf{0}^{k+2} \\ &\subseteq (\exists n^1)(\forall n^{x_1})(\exists n^{x_2}) \dots (Q_k n^{x_k}) \text{BPTS}[n^{c(d-x_1-x_2-\dots-x_k)}] && \mathbf{1}^k \mathbf{0}^{k+1} \\ &\subseteq (\exists n^1)(\forall n^{x_1}) \dots (Q_k n^{x_k}) \text{BPTS}[n^{c x_k}] \\ &\subseteq (\exists n^1)(\forall n^{x_1}) \dots (Q_{k-1} n^{x_{k-1}}) \text{BPTS}[n^{c^2 x_k}] && \mathbf{1}^k \mathbf{0}^{k+1} \\ &\subseteq (\exists n^1)(\forall n^{x_1}) \dots (Q_{k-1} n^{x_{k-1}}) \text{BPTS}[n^{c^2 x_{k-1}}] \\ &\dots && \mathbf{1}^k \mathbf{0}^{k+1} \mathbf{0} \\ &\subseteq (\exists n^1) \text{BPTS}[n^{c^2 x_1}] \\ &\subseteq \text{BPTS}[n^{c^3 x_1}] && \mathbf{1}^k \mathbf{0}^{k+1} \mathbf{0} \\ &\subseteq \text{BPTS}[n^d] \end{aligned}$$

In order for all of the above inclusions to hold, we take

$$x_1 := \frac{d}{c^3}, \quad x_i := (c(1-\epsilon))^{1-i} \cdot x_1 = \frac{d(1-\epsilon)^{1-i}}{c^{i+2}}$$

for  $\epsilon := \frac{1}{k}$ . This automatically satisfies all the constraints except for the one corresponding to the third line, which requires

$$\begin{aligned}
& c(d - x_1 - x_2 - \cdots - x_k) \leq cx_k \\
& \iff 1 - \frac{1}{c^3} \left( \frac{1-\epsilon}{c} \right)^{k-1} - \frac{1}{c^3} \sum_{i=0}^{k-1} \left( \frac{1-\epsilon}{c} \right)^i < 0 \\
& \iff 1 - \frac{1}{c^3} \left( \frac{1-\epsilon}{c} \right)^{k-1} - \frac{1}{c^3} \frac{1 - \frac{1-\epsilon}{c^k}}{1 - \frac{1-\epsilon}{c}} < 0.
\end{aligned}$$

As  $k \rightarrow \infty$ , several terms vanish. We are left with

$$1 - \frac{1}{c^3(1 - \frac{1}{c})} < 0 \iff c^3 - c^2 - 1 < 0.$$

The largest root is at  $c \approx 1.466$ . ◀

When we are allowed to introduce quantum operations and use Lemma 30, we can do slightly better than this.

► **Theorem 36.**  $\exists \cdot \text{BQTIME}[n] \not\subseteq \text{TS}[n^c]$  for  $c < 1.5$ .

The second part of Main Theorem 3 is an immediate corollary of the previous theorem.

**Proof.** (Sketch) Let  $d > 1.5$ . We have the following sequence of containments:

$$\begin{aligned}
\text{BPTS}[n^d] & \subseteq (\exists n^1)(\forall n^{\frac{2d}{3}})(\exists \frac{2d}{3} \log n)^1 \text{TS}[n^{\frac{d}{3}}] \\
& \subseteq (\exists n^1)(\forall n^{\frac{2d}{3}}) \text{BQTIME}[n^{\frac{2d}{3}}] \\
& \subseteq (\exists n^1) \text{BPTS}[n^{\frac{2cd}{3}}],
\end{aligned}$$

where we've used Lemma 29 in the second inclusion. When  $c < 1.5$ , we have  $\frac{2c}{3} < 1$ . Thus, we can repeat this procedure until we derive the inclusion

$$\begin{aligned}
\text{BPTS}[n^d] & \subseteq (\exists n^1) \text{BPTS}[n^{\frac{d}{c}}] \\
& \subseteq \text{BPTS}[n^{d'}]
\end{aligned}$$

for  $d' < d$ . To get a contradiction, we can apply the ideas of Lemma 34 modulo some additional technical details that we relegate to the full version. ◀

---

## References

- 1 Scott Aaronson and Avi Wigderson. Algebrization: A new barrier in complexity theory. *ACM Trans. Comput. Theory*, 1(1), February 2009. doi:10.1145/1490270.1490272.
- 2 Sanjeev Arora and Boaz Barak. *Computational complexity: A modern approach*. Cambridge University Press, Cambridge, 2009. doi:10.1017/CB09780511804090.
- 3 Theodore Baker, John Gill, and Robert Solovay. Relativizations of the  $\mathcal{P} = ?\mathcal{NP}$  question. *SIAM J. Comput.*, 4(4):431–442, 1975. doi:10.1137/0204037.
- 4 Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. Recursive composition and bootstrapping for SNARKS and proof-carrying data. In *STOC'13—Proceedings of the 2013 ACM Symposium on Theory of Computing*, pages 111–120. ACM, New York, 2013. doi:10.1145/2488608.2488623.
- 5 Michel Boyer, Gilles Brassard, Peter Høyer, and Alain Tapp. Tight bounds on quantum searching. *Fortschritte der Physik: Progress of Physics*, 46(4-5):493–505, 1998.
- 6 Sergey Bravyi, David Gosset, and Robert König. Quantum advantage with shallow circuits. *Science*, 362(6412):308–311, 2018.

- 7 Samuel R Buss and Ryan Williams. Limits on alternation trading proofs for time-space lower bounds. *computational complexity*, 24(3):533–600, 2015.
- 8 Scott Diehl. Lower bounds for swapping arthur and merlin. In *APPROX-RANDOM*, 2007.
- 9 Scott Diehl. Lower bounds for swapping arthur and merlin. In Moses Charikar, Klaus Jansen, Omer Reingold, and José D. P. Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 449–463, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.
- 10 Scott Diehl and Dieter van Melkebeek. Time-space lower bounds for the polynomial-time hierarchy on randomized machines. *SIAM J. Comput.*, 36(3):563–594, 2006. doi:10.1137/050642228.
- 11 Lance Fortnow, Richard J. Lipton, Dieter van Melkebeek, and Anastasios Viglas. Time-space lower bounds for satisfiability. *J. ACM*, 52(6):835–865, 2005. doi:10.1145/1101821.1101822.
- 12 Lov K Grover. A fast quantum mechanical algorithm for database search. *arXiv preprint*, 1996. arXiv:quant-ph/9605043.
- 13 Yael Tauman Kalai, Omer Paneth, and Lisa Yang. How to delegate computations publicly. In *STOC’19—Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 1115–1124. ACM, New York, 2019. doi:10.1145/3313276.3316411.
- 14 Yael Tauman Kalai, Omer Paneth, and Lisa Yang. Delegation with updatable unambiguous proofs and ppad-hardness. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology – CRYPTO 2020*, pages 652–673, Cham, 2020. Springer International Publishing.
- 15 Ravindran Kannan. Towards separating nondeterminism from determinism. *Math. Systems Theory*, 17(1):29–45, 1984. doi:10.1007/BF01744432.
- 16 Clemens Lautemann. BPP and the polynomial hierarchy. *Inform. Process. Lett.*, 17(4):215–217, 1983. doi:10.1016/0020-0190(83)90044-3.
- 17 Dylan M. McKay and Richard Ryan Williams. Quadratic time-space lower bounds for computing natural functions with a random oracle. In *10th Innovations in Theoretical Computer Science*, volume 124 of *LIPICs. Leibniz Int. Proc. Inform.*, pages Art. No. 56, 20. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2019.
- 18 V. Nepomnjascii. Rudimentary predicates and turing calculations. *Doklady Mathematics*, 11:1462–1465, 1970.
- 19 Michael A. Nielsen and Isaac L. Chuang. *Quantum computation and quantum information*. Cambridge University Press, Cambridge, 2000.
- 20 Noam Nisan.  $RL \subseteq SC$ . *Comput. Complexity*, 4(1):1–11, 1994. doi:10.1007/BF01205052.
- 21 Ran Raz and Avishay Tal. Oracle separation of BQP and PH. In *STOC’19—Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 13–23. ACM, New York, 2019. doi:10.1145/3313276.3316315.
- 22 Alexander A. Razborov and Steven Rudich. Natural proofs. *J. Comput. Syst. Sci.*, 55(1):24–35, 1997. doi:10.1006/jcss.1997.1494.
- 23 Dieter van Melkebeek and Thomas Watson. Time-space efficient simulations of quantum computations. *Theory Comput.*, 8:1–51, 2012. doi:10.4086/toc.2012.v008a001.
- 24 Adam Bene Watts, Robin Kothari, Luke Schaeffer, and Avishay Tal. Exponential separation between shallow quantum circuits and unbounded fan-in shallow classical circuits. In Moses Charikar and Edith Cohen, editors, *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pages 515–526. ACM, 2019. doi:10.1145/3313276.3316404.
- 25 R. Ryan Williams. Time-space tradeoffs for counting NP solutions modulo integers. In *In Proceedings of the 22nd IEEE Conference on Computational Complexity*, pages 70–82. IEEE, 2007.
- 26 Ryan Williams. Inductive time-space lower bounds for sat and related problems. *Journal of Computational Complexity*, 15:433–470, 2006. doi:10.1007/s00037-007-0221-1.
- 27 Ryan Williams. Alternation-trading proofs, linear programming, and lower bounds. *ACM Trans. Comput. Theory*, 5(2):Art. 6, 49, 2013. doi:10.1145/2493246.2493249.

# Online Search with a Hint

Spyros Angelopoulos 

Sorbonne Université, CNRS, Laboratoire d'informatique de Paris 6, LIP6, 75252 Paris, France

<http://lip6.fr/Spyros.Angelopoulos>

[spyros.angelopoulos@lip6.fr](mailto:spyros.angelopoulos@lip6.fr)

---

## Abstract

The *linear search* problem, informally known as the *cow path* problem, is one of the fundamental problems in search theory. In this problem, an immobile target is hidden at some unknown position on an unbounded line, and a mobile searcher, initially positioned at some specific point of the line called the *root*, must traverse the line so as to locate the target. The objective is to minimize the worst-case ratio of the distance traversed by the searcher to the distance of the target from the root, which is known as the *competitive ratio* of the search.

In this work we study this problem in a setting in which the searcher has a *hint* concerning the target. We consider three settings in regards to the nature of the hint: i) the hint suggests the exact position of the target on the line; ii) the hint suggests the direction of the optimal search (i.e., to the left or the right of the root); and iii) the hint is a general  $k$ -bit string that encodes some information concerning the target. Our objective is to study the *Pareto*-efficiency of strategies in this model. Namely, we seek optimal, or near-optimal tradeoffs between the searcher's performance if the hint is correct (i.e., provided by a trusted source) and if the hint is incorrect (i.e., provided by an adversary).

**2012 ACM Subject Classification** Theory of computation → Online algorithms

**Keywords and phrases** Search problems, searching on the line, competitive analysis, predictions

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.51

**Related Version** A full version of this paper is available at <https://arxiv.org/pdf/2008.13729.pdf>.

**Funding** *Spyros Angelopoulos*: Research benefited from the support of the FMJH Program PGMO and from the support to this program from EDF-THALES-ORANGE. Also partially funded by the grant ANR-19-CE48-0016 from the French National Research Agency (ANR).

**Acknowledgements** I am thankful to Thomas Lidbetter for his comments on an early version of this paper.

## 1 Introduction

Searching for a target is a common task in everyday life, and an important computational problem with numerous applications. Problems involving search arise in such diverse areas as drilling for oil in multiple sites, the forest service looking for missing backpackers, search-and-rescue operations in the open seas, and navigating a robot between two points on a terrain. All these problems involve a mobile *searcher* which must locate an immobile *target*, often also called *hider*, that lies in some unknown point in the *search domain*, i.e., the environment in which the search takes place. The searcher starts from some initial placement within the domain, denoted by  $O$ , which we call the *root*. There is, also, some underlying concept of quality of search, in the sense that we wish, in informal terms, for the searcher to be able to locate the target as efficiently as possible.

One of the simplest, yet fundamental search problems is searching on an infinite line that is unbounded both to the left and to the right of the root. In this problem, which goes back to Bellman [17] and Beck and Newman [10], the objective is to find a search strategy that



© Spyros Angelopoulos;

licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 51; pp. 51:1–51:16

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

minimizes the *competitive ratio* of search. More precisely, let  $S$  denote the *search strategy*, i.e., the sequence of moves that the searcher performs on the line. Given a target  $t$ , let  $d(S, t)$  denote the total distance that the searcher has traveled up to the time it locates the target, and  $d(t)$  the distance of  $t$  from  $O$ . We define the competitive ratio of  $S$  as

$$\text{cr}(S) = \sup_t \frac{d(S, t)}{d(t)}.$$

A strategy of minimum competitive ratio is called *optimal*. The problem of optimizing the competitive ratio of search on the line is known as the *linear search* problem (mostly within Mathematics and Operations Research), but is also known in Computer Science as the *cow path* problem.

It has long been known that the optimal (deterministic) competitive ratio of linear search is 9 [15], and is derived by a simple *doubling* strategy. Specifically, let the two semi-infinite branches of the line be labeled with 0, 1 respectively. Then in iteration  $i$ , with  $i \in \mathbb{N}$ , the searcher starts from  $O$ , traverses branch  $i \bmod 2$  to distance  $2^i$ , and returns to the root.

Linear search, and its generalization, the  $m$ -ray search problem, in which the search domain consists of  $m$  semi-infinite branches have been studied in several settings. Substantial work on linear search was done in the '70s and '80s predominantly by Beck and Beck, see e.g., [11, 16, 12, 13, 14]. Gal showed that a variant of the doubling strategy is optimal for  $m$ -ray search [25, 26]. These results were later rediscovered and extended in [9].

Other related work includes the study of randomization [41] and [30]; multi-searcher strategies [35]; searching with turn cost [23, 3]; the variant in which some probabilistic information on the target is known [27, 28]; the related problem of designing *hybrid algorithms* [29]; searching with an upper bound on the distance of the target from the root [34] and [20]; fault tolerant search [22, 33]; and performance measures beyond the competitive ratio [31, 38, 4]. Competitive analysis has been applied beyond the linear and star search, for example in searching within a graph [32, 24, 6].

## 1.1 Searching with a hint

Previous work on competitive analysis of deterministic search strategies has mostly assumed that the searcher has no information about the target, whose position is adversarial to the search. In practice, however, we expect that the searcher may indeed have some information concerning the target. For instance, in a search-and-rescue mission, there may be some information on the last sighting of the missing person, or the direction the person had taken when last seen. The question then is: how can the searcher leverage such information, and to what possible extent?

If the hint comes from a source that is trustworthy, that is, if the hint is guaranteed to be correct, then the performance of search can improve dramatically. For example, in our problem, if the hint is the branch on which the target lies, then the optimal search is to explore that branch until the target is found, and the competitive ratio is 1. There is, however, an obvious downside: if the hint is incorrect, the search may be woefully inefficient since the searcher will walk eternally on the wrong branch, and the competitive ratio in this case is unbounded.

We are thus interested in analyzing the efficiency of search strategies in a setting in which the hint may be compromised. To this end, we first need to define formally the concept of the hint, as well as an appropriate performance measure for the search strategy. In general, the hint  $h$  is a binary string of size  $k$ , where the  $i$ -th bit is a response to a *query*  $Q_i$ . For example, one can define a single query  $Q$  as “Is the target within distance at most 100 from



$O?$ ” and a one-bit hint, so that the hint answers a range query. For another example, if  $Q$  = “Is the target to the left or to the right of  $O$ ?”, then a 1-bit hint informs the searcher about the direction it should pursue. From the point of view of upper bounds (positive results), we are interested in settings in which the queries and the associated hints have some natural interpretation, such as the ones given above. From the point of view of lower bounds (impossibility results), we are interested on the limitations of general  $k$ -bit hint strings which may be associated with *any* query, as we will discuss in more detail later.

Concerning the second issue, namely evaluating the performance of a search strategy  $S$  with a hint  $h$ , note first that  $S$  is a function of  $h$ . We will analyze the competitiveness of  $S(h)$  in a model in which the competitive ratio is not defined by a single value, but rather by a pair  $(c_{S,h}, r_{S,h})$ . The value  $c_{S,h}$  describes the competitive ratio of  $S(h)$  assuming that  $h$  is *trusted*, and thus guaranteed to be correct. The value  $r_{S,h}$  describes the competitive ratio of  $S(h)$  when the hint is given by an *adversarial* source. More formally, we define

$$c_{S,h} = \sup_t \inf_h \frac{d(S(h), t, h)}{d(t)}, \quad \text{and} \quad r_{S,h} = \sup_t \sup_h \frac{d(S(h), t, h)}{d(t)}, \quad (1)$$

where  $d(S(h), t, h)$  denotes the distance traversed in  $S(h)$  for locating a target  $t$  with a hint  $h$ . We will call  $c_{S,h}$  the *consistency* of  $S(h)$ , and  $r_{S,h}$  the *robustness* of  $S(h)$ . To simplify notation, we will often write  $S$  instead of  $S(h)$  when it is clear from context that we refer to a strategy with a hint  $h$ .

For example, if the hint  $h$  is the branch on which the target lies, then the strategy that always trusts the hint is  $(1, \infty)$  competitive, whereas the strategy that ignores the hint entirely is  $(9, 9)$ -competitive. Our objective is then to find strategies that are provably *Pareto-optimal* or *Pareto-efficient* in this model, and thus identify the strategies with the best tradeoff between robustness and consistency.

Our model is an adaptation, to search problems, of the untrusted advice framework for online algorithms proposed by Angelopoulos *et al.* [5]. In their work, the online algorithm is given some additional information, or *advice* which may, or may not be correct. To the best of our knowledge, our setting is a first attempt to quantify, in an adversarial setting, the impact of general types of *predictions* in search games. It is also in line with recent advances on improving the performance of online algorithms using predictions, such as the work of Lykouris and Vassilvitskii [36], who introduced the concepts of consistency and robustness in the context of paging, and the work of Purohit *et al.* [39], who applied it to general online problems. Our framework for the  $k$ -bit hint is also related to other work in Machine Learning, such as clustering with  $k$  noisy queries, e.g., the work of Mazumdar and Saha [37].

It should be emphasized that there is previous work that has studied the impact of specific types of hints on the performance of search strategies, such as bounds on the maximum or the minimum distance of the target from the root, e.g., [34, 20, 27]. However, the hint in these works is always assumed to be trusted and correct.

## 1.2 Contribution

In this work we study the power of limitations of linear search with hints. Let  $r \geq 9$  be a parameter that in general will denote the robustness of a search strategy, and let  $b_r$  be defined as

$$b_r = \frac{\rho_r + \sqrt{\rho_r^2 - 4\rho_r}}{2}, \quad \text{where } \rho_r = (r - 1)/2.$$

We consider the following classes of hints:

- *The hint is the position of the target.* Here, the hint describes the exact location of the target on the line: its distance from  $O$ , along with the branch (0 or 1) on which it lies. We present a strategy that is  $(\frac{b_r-1}{b_r+1}, r)$ -competitive, and we prove it is Pareto-optimal.
- *The hint is the branch on which the target lies.* Here, the hint is information on whether the searcher is to the left or to the right of the root. We present a strategy that, given parameters  $b > 1$  and  $\delta \in (0, 1)$ , has consistency  $c = 1 + 2 \cdot (\frac{b^2}{b^2-1} + \delta \frac{b^3}{b^2-1})$ , and robustness  $r = 1 + 2 \cdot (\frac{b^2}{b^2-1} + \frac{1}{\delta} \frac{b^3}{b^2-1})$ . Again, we prove that this strategy is Pareto-optimal.
- *The hint is a general  $k$ -bit string.* In the previous settings, the hint is a single bit, which answers the corresponding query. Here we address the question: how powerful can be a single-bit hint, or more generally a  $k$ -bit hint? In other words, how powerful can  $k$  binary queries be for linear search? We give several upper and lower bounds on the competitiveness of strategies in this setting. First, we look at the case of a single-bit hint. Here, we give a 9-robust strategy that has consistency at most  $1 + 4\sqrt{2}$ , whereas we show that no 9-robust strategy can have consistency less than 5, for *any* associated query. For general robustness  $r$ , we give upper and lower bounds that apply to some specific, but broadly used class of strategies, including *geometric* strategies (see Section 2 for a definition and Theorem 10 for the statement of the result). For general  $k$ , and for a given  $r \geq 9$ , we give an  $r$ -robust strategy whose consistency decreases rapidly as function of  $k$  (Proposition 11).

In terms of techniques, for the first setting described above (in which the hint is the position of the target), the main idea is to analyze a geometric strategy with “large” base, namely  $b_r$ , for  $r \geq 9$ . The technical difficulty here is the lower bound; to this end, we prove a lemma that shows, intuitively, that for any  $r$ -robust strategy, the search length of the  $i$ -th iteration cannot be too big compared to the previous search lengths (Lemma 2). This technical result may be helpful in more broad settings (e.g., we also apply it in the setting in which the advice is a general  $k$ -bit string).

Concerning the second setting, in which the hint describes the branch, we rely on tools developed by Schuierer [40] for lower-bounding the performance of search strategies; more precisely on a theorem for lower-bounding the supremum of a sequence of functionals. But unlike [40], we use the theorem in a *parameterized* manner, that allows us to express the tradeoffs between the consistency and the robustness of a strategy, instead of their average.

Concerning the third, and most general setting, our upper bounds (i.e., the positive results) come from a strategy that has a natural interpretation: it determines a partition of the infinite line into  $2^k$  subsets, and the hint describes the partition in which the target lies. The lower bounds (negative results) come from information-theoretic arguments, as is typical in the field of *advice complexity* of online algorithms (see, e.g., the survey [21]).

The broader objective of this work is to initiate the study of search games with some limited, but potentially untrusted information concerning the target. As we will show, the problem becomes challenging even in a simple search domain such as the infinite line. The framework should be readily applicable to other search games, and the analysis need not be confined to the competitive ratio, or to worst-case analysis. For example, search games in bounded domains are often studied assuming a probability distribution on the target, with the objective to minimize the expected search time (for several such examples see the book [1]). However, very little work has addressed the setting in which the searcher may have access to hints, such as the *High-Low* search games described in Section 5.2 of [1], in which a searcher wants to locate a hider on the unit interval by a sequence of guesses. Again, our model is applicable, in that one would like to find the best tradeoff on the expected time to locate the target assuming a trusted or untrusted hint.

## 2 Preliminaries

In the context of searching on the line, a search strategy  $X$  can be defined as an infinite set of pairs  $(x_i, s_i)$ , with  $i \in \mathbb{N}$ ,  $x_i \in \mathbb{R}_{\geq 1}$  and  $s_i \in \{0, 1\}$ . We call  $i$  an *iteration* and  $x_i$  the *length* of the  $i$ -th search *segment*. More precisely, in the  $i$ -th iteration, the searcher starts from the root  $O$ , traverses branch  $s_i$  up to distance  $x_i$  from  $O$ , then returns to  $O$ . It suffices to focus on strategies for which  $x_{i+2} \geq x_i$ , i.e., in any iteration the searcher always searches a new part of the line. We will sometimes omit the  $s_i$ 's from the definition of the strategy, if the direction is not important, i.e., the searcher can start by moving either to the left or to the right of  $O$ . In this case, there is the implicit assumption that  $s_i$  and  $s_{i+1}$  have complementary parities, since any strategy that revisits the same branch in consecutive iterations can be transformed to another strategy that is no worse, and upholds the assumption. We make the standing assumption that the target lies within distance at least a fixed value, otherwise every strategy has unbounded competitive ratio. In particular, we will assume that  $t$  is such that  $d(t) \geq 1$ .

Given a strategy  $X = (x_0, x_1, \dots)$  (which we will denote by  $X = (x_i)$ , for brevity), its competitive ratio is given by the expression

$$\text{cr}(X) = 1 + 2 \sup_{i \geq 0} \frac{\sum_{j=1}^i x_j}{x_{i-1}}, \quad (2)$$

where  $x_{-1}$  is defined to be equal to 1. This expression is obtained by considering all the worst-case positions of the target, namely immediately after the turn point of the  $i$ -th segment (see e.g., [40]).

Geometric sequences are important in search problems, since they often lead to efficient, or optimal strategies (see, e.g., Chapters 7 and 9 in [1]). We call the search strategy  $G_b = (b^i)$  *geometric with base  $b$* . From (2), we obtain that

$$\text{cr}(G_b) = 1 + 2 \frac{b^2}{b-1}. \quad (3)$$

For example, for the standard doubling strategy in which  $x_i = 2^i$ , hence  $b = 2$ , the above expression implies a competitive ratio of 9.

For any  $r \geq 4$  define  $\rho_r$  to be such that  $r = 1 + 2\rho_r$ , thus  $\rho_r = (r-1)/2$ . Moreover, from (3) and the definition of  $b_r$ , we have that  $\text{cr}(G_{b_r}) = r$ .

In the context of searching with a hint, we will say that a strategy is  $(c, r)$ -competitive if it has consistency at most  $c$  and robustness at most  $r$ ; equivalently we say that the strategy is  $c$ -consistent and  $r$ -robust. Clearly, an  $r$ -robust strategy gives rise to a strategy with no hints, and with competitive ratio at most  $r$ .

We conclude with some definitions that will be useful in Section 4. Let  $X = (x_0, x_1, \dots)$  denote a sequence of positive numbers. We define  $\alpha_X$  as

$$\alpha_X = \overline{\lim}_{n \rightarrow \infty} x_n^{1/n}.$$

We also define as  $X^{+i}$  the subsequence of  $X$  starting at  $i$ , i.e.,  $X^{+i} = (x_i, x_{i+1}, \dots)$ . Last, we define the sequence  $G_b(\gamma_0, \dots, \gamma_{n-1})$  as

$$G_b(\gamma_0, \dots, \gamma_{n-1}) = (\gamma_0, \gamma_1 a, \gamma_2 a^2, \dots, \gamma_{n-1} a^{n-1}, \gamma_0 a^n, \gamma_1 a^{n+1}, \dots).$$

### 3 Hint is the position of the target

In this section we study the setting in which the hint is related to the exact position of the target. Namely, the hint  $h$  describes the distance  $d(t)$  of the target  $t$  from the root, as well as the branch on which it hides. For any  $r \geq 9$ , we will give a strategy that is  $(\frac{b_r+1}{b_r-1}, r)$ -competitive. Moreover, we will show that this is Pareto-optimal. We begin with the upper bound.

► **Theorem 1.** *For any  $r \geq 9$  there exists a  $(\frac{b_r+1}{b_r-1}, r)$ -competitive strategy for linear search in which the hint is the position of the target.*

**Proof.** From the hint  $h$ , we have as information the distance  $d(t)$  as well as the branch  $t$  on which the target  $t$  lies; without loss of generality, suppose that this branch is the branch 0. Recall that this information may or may not be correct, and the searcher is oblivious to this.

Consider the geometric strategy  $G_{b_r} = (b_r^i)$ , with  $i \in \mathbb{N}$ , and recall that  $G_{b_r}$  is  $r$ -robust (as discussed in Section 2). There must exist an index  $j_t$  such that  $b_r^{j_t-2} < d(t) \leq b_r^{j_t}$ . Define  $\lambda = b_r^{j_t}/d(t) \geq 1$ , and let  $G'$  denote the strategy  $G' = (\{\frac{1}{\lambda}b_r^i, s_i\})$ , where the  $s_i$ 's are defined such that that  $s_{i+1} \neq s_i$ , for all  $i$ , and  $s_{j_t} = 0$ .

In words,  $G'$  is obtained by “shrinking” the search lengths of  $G_{b_r}$  by a factor equal to  $\lambda$ , and by choosing the right parity of branch for starting the search, in a way that, if the hint is trusted, then in  $G'$  the searcher will locate the target right as it is about to turn back to  $O$  at the end of the  $j_t$ -th iteration.

Since  $G_{b_r}$  is  $r$ -robust, so is the scaled-down strategy  $G'$ ; this is because the worst-case competitive ratio is attained for targets hiding right after the turn points of the segments. It remains then to bound the consistency  $c_{G'}$  of  $G'$ . Suppose that the hint is trusted. We have that

$$d(G', t) = \frac{1}{\lambda} \left( 2 \sum_{i=0}^{j_t-1} b_r^i + b_r^{j_t} \right),$$

and since  $d(t) = b_r^{j_t}/\lambda$  we can bound  $c_{G'}$  from above by

$$\frac{d(G', t)}{d(t)} = 1 + 2 \frac{b_r^{j_t} - 1}{b_r^{j_t}(b_r - 1)} \leq 1 + \frac{2}{b_r - 1} = \frac{b_r + 1}{b_r - 1}.$$

We conclude that  $G'$  is  $(\frac{b_r+1}{b_r-1}, r)$ -competitive. ◀

Next, we will show that the strategy of Theorem 1 is Pareto-optimal. To this end, we will need a technical lemma concerning the segment lengths of any  $r$ -robust strategy.

► **Lemma 2.** *For any  $r$ -robust strategy  $X = (x_i)$ , it holds that*

$$x_i \leq \left( b_r + \frac{b_r}{i+1} \right) x_{i-1},$$

for all  $i \geq 1$ , where  $x_{-1}$  is defined to be equal to 1.

We obtain a useful corollary concerning the sum of the first  $i-1$  search lengths of an  $r$ -robust strategy.

► **Corollary 3.** *For any  $r$ -robust strategy  $X = (x_i)$ , it holds that*

$$\sum_{j=0}^{i-1} x_j \geq \frac{x_i}{1 + \frac{1}{i+1}} \left( \frac{b_r}{b_r - 1} - \frac{i+2}{i+1} \right),$$

and for every  $\epsilon \in (0, 1]$ , there exists  $i_0$  such that for all  $i > i_0$ ,  $\sum_{j=0}^{i-1} x_j \geq (\frac{1}{b_r-1} - \epsilon)x_i$ .

We can now show a lower bound on the competitiveness of every strategy that matches the upper bound of Theorem 1.

► **Theorem 4.** *For every  $(c, r)$ -competitive strategy for linear search in which the hint is the position of the target, it holds that  $c \geq \frac{b_r+1}{b_r-1} - \epsilon$ , for any  $\epsilon > 0$ .*

**Proof.** Let  $X = (x_i)$  denote an  $r$ -robust strategy, with a hint that specifies the position of a target  $t$ . Suppose that  $X$  locates the target at the  $j_t$ -th iteration. We have that

$$c = \frac{d(X, t)}{d(t)} = \frac{2 \sum_{i=0}^{j_t-1} x_i + d(t)}{d(t)} \geq \frac{2 \sum_{i=0}^{j_t-1} x_i + x_{j_t}}{x_{j_t}} = 1 + 2 \frac{\sum_{i=0}^{j_t-1} x_i}{x_{j_t}}.$$

Note that the target  $t$  can be chosen to be arbitrarily far from  $O$ , which means that  $j_t$  can be unbounded (otherwise the strategy would not have bounded robustness). From Corollary 3 this implies that  $\sum_{i=0}^{j_t-1} x_i$  can be arbitrarily close to  $x_{j_t} \frac{1}{b_r-1}$ , and therefore  $c$  is arbitrarily close to  $1 + 2 \frac{1}{b_r-1} = \frac{b_r+1}{b_r-1}$ , which concludes the proof. ◀

#### 4 Hint is the direction of search

In this section we study the setting in which the hint is related to the *direction* of the search. More precisely, the hint is a single bit that dictates whether the target is to the left or to the right of the root  $O$ . Again, we are interested in Pareto-optimal strategies with respect to competitiveness: namely, for any fixed  $r \geq 9$ , what is the smallest  $c$  such that there exist  $(c, r)$ -competitive strategies?

A related problem was studied by Schuierer [40], which is called *biased search*. One defines the *left* and *right* competitive ratios, as the competitive ratio of a search, assuming that the target hides to the left of the root, or to the right of the root, respectively. However, the searcher does not know the target's branch. Of course we know that the maximum of the left and the right competitive ratios is at least 9 (and for the doubling strategy, this is tight). [40] shows that for any search strategy on the line (not necessarily 9-robust), the *average* of the left and the right competitive ratios is at least 9. At first glance, one may think that this could be an unsurprising, and perhaps even trivial result; however this is not the case. The proof in [40] is not straightforward, and relies in a generalization of a theorem of [25] which lower bounds the supremum of a sequence of functionals by the supremum of much simpler, geometric functionals. We will discuss this theorem shortly.

The problem studied in [40] is related to our setting: the left and right competitive ratios correspond to the consistency  $c$  and the robustness  $r$  of the strategy. Hence from [40] we know that  $c + r \geq 18$ . However, there is a lot of room for improvement. In this section we will show a much stronger tradeoff between  $c$  and  $r$ , and we will further prove that it is tight. For example, we will show that for any  $(c, r)$ -competitive strategy, if  $c$  approaches 5 from above, then  $r$  approaches infinity (in contrast, in this case, the lower bound of [40] yields  $r \geq 13$ ). In fact, we will show that  $c + r$  is minimized when  $c = r = 9$ . To this end, we will apply a parameterized analysis based on Schuierer's approach. We begin with the upper bound, by analyzing a specific strategy.

► **Theorem 5.** *For every  $b \geq 1$ , and  $\delta \in (0, 1]$ , there is a  $(c, r)$ -competitive strategy for linear search with the hint being the direction of search, in which*

$$c = 1 + 2 \cdot \left( \frac{b^2}{b^2 - 1} + \delta \frac{b^3}{b^2 - 1} \right) \text{ and } r = 1 + 2 \cdot \left( \frac{b^2}{b^2 - 1} + \frac{1}{\delta} \frac{b^3}{b^2 - 1} \right).$$

**Proof.** Suppose, without loss of generality, that the hint points to branch 0. Consider a strategy  $X = (\{x_i, i \bmod 2\})$ , which starts with branch 0, and alternates between the two branches. This strategy has consistency and robustness given by the following expressions, as a consequence of (2):

$$c = 1 + 2 \cdot \sup_{k \geq 0} \left\{ \frac{\sum_{i=0}^{2k+1} x_i}{x_{2k}} \right\} \quad \text{and} \quad r = 1 + 2 \cdot \sup_{k \geq 0} \left\{ \frac{\sum_{i=0}^{2k} x_i}{x_{2k-1}} \right\}, \quad (4)$$

where  $x_{-1}$  is defined to be equal to 1.

In addition, the search lengths of  $X$  are defined by

$$x_i = b^i, \text{ if } i \text{ even and } x_i = \delta b^i, \text{ if } i \text{ is odd,}$$

where we require that  $b > 1$ , and  $\delta \in (0, 1]$ . Note that  $X$  is “biased” with respect to branch 0, which makes sense since the hint points to that branch.

Substituting these values into (4), we obtain that

$$\begin{aligned} c &= 1 + 2 \cdot \sup_{k \geq 0} \left\{ \frac{\sum_{i=0}^k b^{2i}}{b^{2k}} + \delta \frac{\sum_{i=0}^k b^{2i+1}}{b^{2k}} \right\} = 1 + 2 \cdot \sup_{k \geq 0} \left\{ \frac{b^{2(k+1)} - 1}{(b^2 - 1)b^{2k}} + \delta \frac{b^{2k+3} - 1}{(b^2 - 1)b^{2k}} \right\} \\ &\leq 1 + 2 \cdot \left( \frac{b^2}{b^2 - 1} + \delta \frac{b^3}{b^2 - 1} \right). \end{aligned}$$

Similarly, we have that

$$\begin{aligned} r &= 1 + 2 \cdot \sup_{k \geq 0} \left\{ \frac{1}{\delta} \frac{\sum_{i=0}^k b^{2i}}{b^{2k-1}} + \frac{\sum_{i=0}^{k-1} b^{2i+1}}{b^{2k-1}} \right\} = 1 + 2 \cdot \sup_{k \geq 0} \left\{ \frac{1}{\delta} \frac{b^{2(k+1)} - 1}{(b^2 - 1)b^{2k-1}} + \frac{b^{2k+1} - 1}{(b^2 - 1)b^{2k-1}} \right\} \\ &\leq 1 + 2 \cdot \left( \frac{1}{\delta} \frac{b^3}{b^2 - 1} + \frac{b^2}{b^2 - 1} \right). \end{aligned} \quad \blacktriangleleft$$

For example, if  $\delta = 1$ , and  $b = 2$ , then Theorem 5 shows that there exists a (9, 9)-competitive strategy. Interestingly, the theorem shows that as the consistency  $c$  approaches 5 from above, the robustness  $r$  of the strategy must approach infinity. This is because the function  $\frac{b^2}{b-1}$  is minimized for  $b = 2$ , and hence for  $c$  to approach 5 from above, it must be that  $b$  approaches 2, and  $\delta$  approaches 0. But then  $\frac{1}{\delta}$  must approach infinity, and so must  $r$ .

We will show that the strategy of Theorem 5 is Pareto-optimal. To this end, we will use the following theorem of [40]. Recall the definitions of  $\alpha_X$ ,  $X^{+i}$  and  $G_a(\gamma_0, \dots, \gamma_{n-1})$  given in Section 2.

► **Theorem 6** (Theorem 1 in [40]). *Let  $p, q$  be two positive integers, and  $X = (x_0, x_1, \dots)$  a sequence of positive numbers with  $\sup_{n \geq 0} x_{n+1}/x_n < \infty$  and  $\alpha_X > 0$ . Suppose that  $F_k$  is a sequence of functionals that satisfy the following properties:*

- (1)  $F_k(X)$  depends only on  $x_0, x_1, \dots, x_{pk+q}$ ,
- (2)  $F_k(X)$  is continuous in every variable, for all positive sequences  $X$ ,
- (3)  $F_k(aX) = F_k(X)$ , for all  $a > 0$ ,
- (4)  $F_k(X + Y) \leq \max(F_k(X), F_k(Y))$ , for all positive sequences  $X, Y$ , and
- (5)  $F_{k+i}(X) \geq F_k(X^{+ip})$ , for all  $i \geq 1$ .

*Then there exist  $p$  positive numbers  $\gamma_0, \gamma_1, \gamma_{p-1}$  such that*

$$\sup_{0 \leq k < \infty} F_k(X) \geq \sup_{0 \leq k < \infty} F_k(G_{\alpha_X}(\gamma_0, \dots, \gamma_{p-1})).$$

We will use Theorem 6 to prove a tight lower bound on the competitiveness of any strategy  $X$ .

► **Theorem 7.** *For every  $(c, r)$ -competitive strategy, there exists  $\alpha > 1$ , and  $\delta \in (0, 1]$  such that  $c \geq 1 + 2 \cdot (\frac{\alpha^2}{\alpha^2-1} + \delta \frac{\alpha^3}{\alpha^2-1})$ , and  $r \geq 1 + 2 \cdot (\frac{\alpha^2}{\alpha^2-1} + \frac{1}{\delta} \frac{\alpha^3}{\alpha^2-1})$ .*

**Proof.** Let  $X = (x_0, x_1, \dots)$  denote a  $(c, r)$ -competitive strategy, and suppose, without loss of generality, that the hint specifies that the target is in the branch labeled 0. There are two cases concerning  $X$ : either the first exploration is on the branch labeled 0, or on the branch labeled 1. Let us assume the first case; at the end, we will argue that the second case follows from a symmetrical argument. As we argued in the proof of Theorem 5, in this case the competitiveness of  $X$  is described by (4). Let us define the functionals

$$C_k = \frac{\sum_{i=0}^{2k+1} x_i}{x_{2k}} \quad \text{and} \quad R_k = \frac{\sum_{i=0}^{2k} x_i}{x_{2k-1}}.$$

Then we have that

$$c = 1 + 2 \cdot \sup_{k \geq 0} C_k \quad \text{and} \quad r = 1 + 2 \cdot \sup_{k \geq 0} R_k. \quad (5)$$

The functional  $C_k$  satisfies the conditions of Theorem 6 with  $p = 2$ , as shown in [40] therefore there exist  $\gamma_0, \gamma_1 > 0$  such that

$$\begin{aligned} \sup_{k \geq 0} C_k &\geq \sup_{k \geq 0} C_k(G_{\alpha_X}(\gamma_0, \gamma_1)) = \sup_{k \geq 0} \frac{\gamma_0 + \gamma_1 \alpha_X + \gamma_0 \alpha_X^2 + \dots + \gamma_0 \alpha_X^{2k} k + \gamma_1 \alpha_X^{2k+1}}{\gamma_0 \alpha_X^{2k}} \\ &= \sup_{k \geq 0} \left\{ \frac{\sum_{i=0}^k \alpha_X^{2i}}{\alpha_X^{2k}} + \frac{\gamma_1}{\gamma_0} \frac{\sum_{i=0}^k \alpha_X^{2i+1}}{\alpha_X^{2k}} \right\}. \end{aligned}$$

If  $\alpha_X \leq 1$ , then the above implies that  $\sup_{k \geq 0} C_k = \infty$  (another way of dismissing this case is that if  $\alpha_X \leq 1$ , then  $X$  is bounded and the two branches are not explored to infinity, as required by any strategy of bounded consistency). We can thus assume that  $\alpha_X > 1$ , and we obtain that

$$\sup_{k \geq 0} C_k \geq \sup_{k \geq 0} \left\{ \frac{\alpha_X^{2k+2} - 1}{(\alpha_X^2 - 1) \alpha_X^{2k}} + \frac{\gamma_1}{\gamma_0} \frac{\alpha_X^{2k+3} - 1}{(\alpha_X^2 - 1) \alpha_X^{2k}} \right\} = \frac{\alpha_X^2}{\alpha_X^2 - 1} + \frac{\gamma_1}{\gamma_0} \frac{\alpha_X^3}{\alpha_X^2 - 1}. \quad (6)$$

We can lower-bound  $r$  using a similar argument, and obtain

$$\sup_{k \geq 0} R_k \geq \sup_{k \geq 0} \left\{ \frac{\gamma_0}{\gamma_1} \frac{\alpha_X^{2k+2} - 1}{(\alpha_X^2 - 1) \alpha_X^{2k-1}} + \frac{\alpha_X^{2k+1} - 1}{(\alpha_X^2 - 1) \alpha_X^{2k-1}} \right\} = \frac{\gamma_0}{\gamma_1} \frac{\alpha_X^3}{\alpha_X^2 - 1} + \frac{\alpha_X^2}{\alpha_X^2 - 1}. \quad (7)$$

Let us define  $\delta = \frac{\gamma_1}{\gamma_0} > 0$ . The result follows then by combining (5), (6) and (7). Note that if we require that  $c \leq r$ , it must be that  $\delta \leq 1$ , since  $\alpha_X > 1$ .

It remains to consider the symmetric case, in which in  $X$ , the first explored branch is branch 1. In this case the analysis is essentially identical: in (5) we substitute  $C_k$  with  $R_k$  and vice versa, in the expressions of  $c$  and  $r$ , and in the resulting lower bounds we require that  $\delta > 1$ . ◀

It is important to note that in the proof of Theorem 7 we used the fact that the values  $\gamma_0$  and  $\gamma_1$  depend only on  $X$  and not on any functionals defined over  $X$ , as follows from the proof of Theorem 6 in [40].



Theorem 7 implies that any  $(c, r)$ -competitive strategy  $X$  is such that

$$c + r \geq 2 + 4 \frac{\alpha_X^2}{\alpha_X^2 - 1} + 2 \left( \delta + \frac{1}{\delta} \right) \frac{\alpha_X^3}{\alpha_X^2 - 1},$$

which is minimized at  $\delta = 1$ , hence  $c + r$  is minimized only if  $c = r = 1 + 2 \frac{\alpha_X^2}{\alpha_X^2 - 1} \geq 9$ . We conclude that the average of a strategy's consistency and robustness (or the average of the left and right competitive ratio, in the terminology of [40]) is minimized only by strategies that are 9-robust.

## 5 Hint is a $k$ -bit string

In this section we study the setting in which the searcher has access to a *hint string* of  $k$  bits. We first consider the case  $k = 1$ . In Section 5.1 we will study the more general case.

It should be clear that even a single-bit hint is quite powerful, and that the setting is non-trivial. For example, the bit can indicate the right direction for search, as discussed in Section 4, but it allows for other possibilities, such as whether the target is at distance at most  $D$  from the root, for some chosen  $D$ . The latter was studied in [27], assuming that the hint is correct. More generally, the hint can induce a partition of the infinite line into two subsets  $L_1$  and  $L_2$ , such that the hint dictates whether the target is hiding on  $L_1$  or  $L_2$ .

We begin with the upper bound, namely we describe a specific search strategy, and the corresponding hint bit (as well as the query which it responds). Consider two strategies of the form

$$X_1 = (b_r^i) \text{ and } X_2 = (b_r^{i+\frac{1}{2}}).$$

Note that both  $X_1$  and  $X_2$  are  $r$ -robust:  $X_1$  is geometric with base  $b_r$ , whereas  $X_2$  is obtained from  $X_1$  by scaling the search lengths by a factor equal to  $b_r^{1/2}$ . We also require that the two strategies start by searching the same branch, hence in every iteration, they likewise search the same branch.

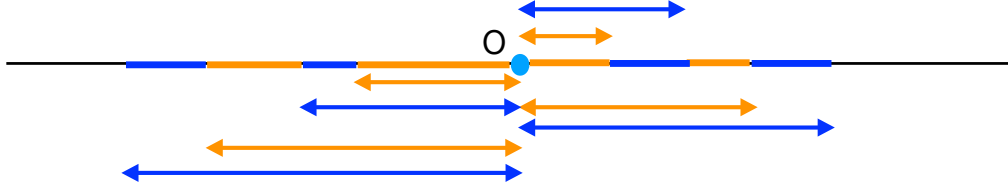
We can now define a strategy  $Z$  with a single bit hint, which indicates whether the searcher should choose strategy  $X_1$  or strategy  $X_2$ . For any given target, one of the two strategies will outperform the other, assuming the hint is trusted. Thus, an equivalent interpretation of the hint is in the form of a partition of the infinite line into two sets  $L_1$  and  $L_2$ , such that if the target is in  $L_i$ , then  $X_i$  is the preferred strategy, with  $i \in [1, 2]$ . See Figure 1 for an illustration.

The following result bounds the performance of this strategy, and its proof will follow as a corollary of a more general theorem concerning  $k$ -bit strings that we show in Section 5.1 (Theorem 11).

► **Proposition 8.** *For given  $r \geq 9$ , the above-defined strategy  $Z$  is  $r$ -robust and has consistency at most  $1 + 2 \frac{a^{3/2}}{a-1}$ , where  $a = b_r$ , if  $r \leq 10$ , and  $a = 3$ , otherwise.*

Note that if  $r = 9$ , then  $Z$  has consistency  $1 + 4\sqrt{2} \approx 6.657$ . For  $r \in [9, 10]$ , the consistency of  $Z$  is decreasing in  $r$ , as one expects. For  $r \geq 10$ , the consistency is  $1 + 3\sqrt{3} \approx 6.196$ .

We now turn our attention to lower bounds. To this end, we observe that a single-bit hint  $h$  has only the power to differentiate between two *fixed* strategies, say  $X = (x_i)$ , and  $Y = (y_i)$ , i.e., two strategies that are not defined as functions of  $h$ . We say that  $Z$  is *determined* by strategies  $X$  and  $Y$ , and the bit  $h$ .



■ **Figure 1** Illustration of strategy  $Z$ , using the first four segments of strategies  $X_1$  and  $X_2$ . Blue (dark) and orange (faded) segments correspond to the search segments of strategies  $X_1$  and  $X_2$ , respectively. The parts of the line in blue (resp. orange) indicate the hiding intervals for the target such that  $X_1$  (resp.  $X_2$ ) is preferred, and thus chosen by the hint.

### Setting up the lower-bound proofs

We give some definitions and notation that will be used in the proofs of Theorems 9 and 10. Let  $Z$  be determined by strategies  $X$  and  $Y$ , and a single-bit hint  $h$ . Let  $C$  denote the lower bound on the consistency of  $Z$  that we wish to show. For given  $i$ , define  $T_X^i = 2 \sum_{j=0}^i x_j + x_{i-1}$ , and similarly for  $T_Y^i$ . Define also  $q = r/C$ .

Note that a searcher that follows strategy  $X$  will turn towards the root at iteration  $i - 1$ , after having explored some branch  $\beta_i \in \{0, 1\}$  up to distance  $x_{i-1}$ . Thus,  $X$  barely misses a target that may be hiding at branch  $\beta_i$ , and at distance  $x_{i-1} + \epsilon$  from  $O$ , with  $\epsilon > 0$  infinitesimally small, and thus requires time  $T_X^i$  to discover it. We will denote this hiding position of a potential target by  $P_i$ . If, on the other hand, the searcher follows  $Y$ , then it can locate a target at position  $P_i$  at a time that may be smaller than  $T_X^i$ ; let  $\tau_i$  denote this time. When strategy  $Y$  locates a target hiding at  $P_i$ , it does so by exploring branch  $\beta_i$  to a length greater than  $x_{i-1}$ . Let  $j_i$  be the iteration at which  $Y$  locates  $P_i$ , thus  $y_{j_i} \geq x_{i-1}$ . Last, let  $Q_i$  denote the position in branch  $\beta_i$  and at distance  $y_{j_i} + \epsilon$  from  $O$ . In words, if a target hides at  $Q_i$ , then strategy  $Y$  barely misses it when executing the search segment  $y_{j_i}$ .

We first show a lower bound on the consistency of 9-robust strategies. In the proof we will not replace all parameters with the corresponding values (e.g., we will sometimes use  $r$  to refer to robustness, instead of the value 9). We do so because the arguments in the proof can be applied to other settings, as will become clear in the proof of Theorem 10.

► **Theorem 9.** *For any  $(c, 9)$ -competitive strategy with single-bit hint, it holds that  $c \geq 5$ .*

**Proof.** We will prove the result by way of contradiction. Let  $C = 5$ , and suppose that there is a strategy  $Z$  of consistency strictly less than  $C$ . Let  $Z$  be determined by two fixed strategies  $X$  and  $Y$ . Both  $X$  and  $Y$  must be  $r$ -robust (i.e., 9-robust), otherwise  $Z$  cannot be  $r$ -robust.

Fix  $i_0 \in \mathbb{N}$ . Suppose first that  $i_0$  is such that for all  $i \geq i_0$ , we have  $\tau_i \geq \frac{1}{q} T_X^i$ . In this case, for a target at position  $P_i$ , defined earlier,  $X$  requires time  $T_X^i$  to locate it, whereas  $Y$  requires time at least  $\tau_i \geq \frac{1}{q} T_X^i$  to locate it, thus the minimum time  $X$  or  $Y$  can locate this target is  $\frac{1}{q} T_X^i$ . Therefore, the consistency of  $Z$  is at least

$$c \geq \sup_{i \geq i_0} \frac{T_X^i}{q \cdot x_{i-1}} = \frac{1}{q} \sup_{i \geq i_0} \frac{T_X^i}{x_{i-1}} \geq \frac{C}{r} \cdot r = C, \quad (8)$$

which is a contradiction. Here, we used crucially the fact that  $\sup_{i \geq i_0} \frac{T_X^i}{x_{i-1}} \geq 9$ , for any 9-robust strategy  $X$  and any  $i_0$ <sup>1</sup>. Specifically, there exists sufficiently large  $i$  such that  $T_X^i$  is arbitrarily close to  $9x_{i-1}$ .

It must then be that  $i_0$  does not obey the property described above, namely for some  $i \geq i_0$  we have that  $\tau_i \leq \frac{1}{q}T_X^i$ . Since  $X$  is  $r = 9$ -robust, it must also be that  $\frac{T_X^i}{x_{i-1}} \leq r$ , as can be seen if a target hides at  $P_i$ . We therefore obtain that

$$\tau_i \leq \frac{1}{q}T_X^i \leq \frac{1}{q} \cdot r \cdot x_{i-1} = Cx_{i-1}. \quad (9)$$

We can also give a lower bound on  $\tau_i$ , as follows. Recall that we denote by  $y_{j_i}$  the segment at which strategy  $Y$  locates a target at position  $P_i$ . For arbitrarily small  $\epsilon > 0$ , we can choose  $i_0$  sufficiently large, which also implies that  $j_i$  can also be sufficiently large (since otherwise  $Y$  would not have finite robustness), so that Corollary 3 applies. To simplify the arguments, in the remainder of the proof we will assume that the corollary applies with  $\epsilon = 0$ ; this has no effect on correctness, since we want to show a lower bound of the form  $C - \delta$  on the consistency, and  $\epsilon$  can be made as small as we want in comparison to  $\delta$ . More precisely, we obtain that

$$\tau_i = 2 \sum_{l=0}^{j_i} y_l + x_{i-1} \geq \frac{2}{b_r - 1} y_{j_i} + x_{i-1}. \quad (10)$$

Combining (9) and (10) we have

$$y_{j_i} \leq \frac{C-1}{2}(b_r - 1)x_{i-1}. \quad (11)$$

In particular, since  $r = 9$  and  $C = 5$ , we have that  $y_{j_i} \leq 2x_{i-1}$ .

Consider now a target at position  $Q_i$ , and recall that this position is at distance infinitesimally larger than  $y_{j_i}$ . We will show that in both  $X$  and  $Y$ , there exists an  $i \geq i_0$  such that the searcher walks distance at least  $C \cdot y_{j_i}$  before reaching this position, which implies that  $Z$  has consistency at least  $C$ , and which yields the contradiction.

Consider first strategy  $Y$ . In this case, the searcher walks distance at least  $T_Y^{j_i}$ , to reach  $Q_i$ , from the definition of  $T_Y$ . Since  $r = 9$ , we know that  $\sup_{i \geq i_0} \frac{T_Y^{j_i}}{y_{j_i}} \geq 9$ , for any  $i_0$ , hence there exists an  $i \geq i_0$  such that the distance walked by the searcher is at least  $r \cdot y_{j_i}$ , and hence at least  $C \cdot y_{j_i}$ .

Consider now strategy  $X$ . In this case, in order to arrive at position  $Q_i$ , the searcher needs to walk distance  $T_X^i$ , then at least an additional distance  $y_{j_i} - x_{i-1}$  to reach  $Q_i$ . Let us denote by  $D_X^i$  this distance. We have

$$\begin{aligned} D_X^i &= T_X^i + y_{j_i} - x_{i-1} \\ &\geq 9x_{i-1} + y_{j_i} - x_{i-1} \quad (\text{From Corollary 3 and since } T_X^i \text{ is arbitrarily close to } 9x_{i-1}) \\ &= 8x_{i-1} + y_{j_i}. \end{aligned} \quad (12)$$

We then bound the ratio  $D_X^i/y_{j_i}$  from below as follows.

$$\begin{aligned} \frac{D_X^i}{y_{j_i}} &\geq \frac{8x_{i-1} + y_{j_i}}{y_{j_i}} = 1 + 8 \frac{x_{i-1}}{y_{j_i}} \\ &\geq 1 + 8 \frac{x_{i-1}}{2 \cdot x_{i-1}} \quad (\text{From the fact that } y_{j_i} \leq 2 \cdot x_{i-1}) \\ &= 5. \end{aligned} \quad (13)$$

We thus conclude that  $C \geq 5$ , which yields the contradiction, and completes the proof. ◀

---

<sup>1</sup> In general, this statement is not immediately true for arbitrary  $r > 9$ .

Showing a lower bound on the consistency, as a function of general  $r > 9$  is quite hard, even for the case of a single-bit hint. The reason is that as  $r$  increases, so does the space of  $r$ -robust strategies. For example, any geometric strategy  $G_b$  has robustness  $r$ , as long as  $b \in [\frac{\rho_r - \sqrt{\rho_r^2 - 4\rho_r}}{2}, \frac{\rho_r + \sqrt{\rho_r^2 - 4\rho_r}}{2}]$ . In what follows we will show a lower bound for a class of strategies which we call *asymptotic*. More precisely, recall the definition of  $T_X^i$ . We call an  $r$ -robust strategy  $S$  *asymptotic* if  $\sup_{i \geq i_0} \frac{T_X^i}{x_{i-1}} = r$ , for all fixed  $i_0$ . In words, in an asymptotic strategy, the worst-case robustness (i.e., the worst case competitive ratio without any hint) can always be attained by targets placements sufficiently far from the root. All geometric strategies, including the doubling strategy, have this property, and this holds for many strategies that solve search problems on the line and the star, such as the ones described in the introduction. Note also that the strategies  $X_1$  and  $X_2$  that determine the strategy  $Z$  in the statement of Proposition 8 are asymptotic, since they are near-geometric. Thus, the lower bound we show in the next theorem implies that in order to substantially improve consistency, one may have to resort to much more complex, and most likely irregular strategies.

► **Theorem 10.** *Let  $Z$  denote a strategy with 1-bit hint which is determined by two  $r$ -robust, asymptotic strategies  $X$  and  $Y$ . Then  $Z$  is  $(c, r)$ -competitive, with  $c \geq 1 + \frac{2b_r}{b_r - 1}$ .*

### 5.1 $k$ -bit hints

Here we consider the general setting in which the hint is a  $k$ -bit string, for some fixed  $k$ . First, we give an upper bound that generalizes Proposition 8. We will adapt an algorithm proposed in [5] for the *online bidding* problem with untrusted advice. Consider  $2^k$  strategies  $X_0, \dots, X_{2^k-1}$ , where

$$X_j = (a^{i + \frac{j}{2^k}})_{i \geq 0},$$

for some  $a$  to be determined later, and where all the  $X_j$  have the same parity: they all search the same branch in their first iteration and, therefore in every iteration as well. Define a strategy  $Z$ , which is determined by  $X_0, \dots, X_{2^k-1}$ , and in which the  $k$ -bit hint  $h$  dictates the index  $j$  of the chosen strategy  $X_j$ . In other words,  $h$  answers the query  $Q_h$  = “which strategy among  $X_0, \dots, X_{2^k-1}$  should the searcher choose?”. An equivalent interpretation is that the statements of the  $X_j$ ’s induce a partition of the line, such that for every given target position, one of the  $X_j$ ’s is the preferred strategy. Thus every bit  $i$  of the hint can be thought, equivalently, as the answer to a partition query  $Q_i$  of the line, i.e., of the form “does the target belong in a subset  $L_i$  of the line or not?”.

► **Theorem 11.** *For every  $r \geq 9$ , the strategy  $Z$  defined above is  $(c, r)$ -competitive with  $c \leq 1 + 2^{\frac{1 + \frac{1}{2^k}}{a-1}}$ , where  $a = b_r$ , if  $\rho_r \leq \frac{(1+2^k)^2}{2^k}$ , and  $a = 1 + 2^k$ , otherwise.*

For example, for  $r = 9$ , we obtain a strategy that is  $(1 + 2^{2 + \frac{1}{2^k}}, 9)$ -competitive. Thus, the consistency decreases rapidly, as function of  $k$ , and approaches 5.

Last, it is easy to see that no 9-robust strategy with hint string of any size can have consistency better than 3. To see this, let  $X$  be any 9-robust strategy, and let  $i_t$  be the iteration in which it locates a target  $t$ . Since  $t$  can be arbitrarily far from  $O$ ,  $i_t$  is unbounded, and thus Corollary 3 applies. We thus have that

$$\frac{d(X, t)}{d(t)} = \frac{\sum_{j=0}^{i_t-1} x_j + d(t)}{d(t)} = 1 + 2 \frac{\sum_{j=0}^{i_t-1} x_j}{d(t)} \geq 1 + 2 \frac{\sum_{j=0}^{i_t-1} x_j}{x_{i_t}} \geq 1 + 2 \frac{(1 - \epsilon)x_{i_t}}{x_{i_t}},$$

thus the consistency of  $X$  cannot be smaller than  $3 - \epsilon$ , for any  $\epsilon$ . The same holds then for any strategy that is determined by any number of 9-robust strategies, and thus for any hint.

## 6 Conclusion

We introduced a model that allows us to analyze the performance of search strategies with some information on the target. As explained in the introduction, the eventual goal is to apply this model to many other search problems, within or beyond competitive analysis, and for deterministic or randomized strategies. A good starting point is the generalization of linear search, namely the  $m$ -ray star problem, which has a long history of research, and has a broad connection to resource allocation problems (see, e.g. [33, 8, 18]). In this context, there are  $m$  tasks and the goal is to allocate resources to each of them, without knowing ahead of time which task will be the crucial one. The hint can be interpreted as some information concerning the instance (e.g., the index of the crucial task, or the time we may have to allocate to each one of the tasks in order to guarantee some good solution).

Another direction is to consider the case in which the hint comes with some error  $\eta$ , as in recent works in Machine Learning [36, 39, 37]. The ultimate goal would be to describe Pareto-efficient solutions in which the competitive ratio of the search also degrades gently as  $\eta$  increases. This is a challenging task, since one will have to strike a balance between the robustness, the consistency, and the competitive ratio, both from the point of upper and lower bounds. But there is a major obstacle as far as linear search is concerned. That is, unlike other online problems, the competitive ratio of linear search is not necessarily an increasing function of the error  $\eta$ . For example, consider the case in which there is a prediction of the position of the target, and suppose that we have a “bad” search strategy, which does not perform well if the error is small (i.e., turns right before discovering the target). However, it is possible that with significant error, the target happens to be found just as the searcher is about to turn and return to  $O$ , in one of the iterations. In this case, the strategy is quite efficient. These are counterintuitive complications that need to be taken into consideration in terms of defining efficiency in this setting.

In very recent work [7], we addressed the above questions in the context of a related problem in AI, namely the problem of *contract scheduling* under the *acceleration ratio*. There are interesting connections between searching on the line and contract scheduling [19, 2], and the techniques we developed and the results we showed in [7] should be readily applicable in searching on the line with noisy predictions.

---

## References

- 1 Steve Alpern and Shmuel Gal. *The theory of search games and rendezvous*. Kluwer Academic Publishers, 2003.
- 2 Spyros Angelopoulos. Further connections between contract-scheduling and ray-searching problems. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1516–1522, 2015.
- 3 Spyros Angelopoulos, Diogo Arsénio, and Christoph Dürr. Infinite linear programming and online searching with turn cost. *Theoretical Computer Science*, 670:11–22, 2017.
- 4 Spyros Angelopoulos, Christoph Dürr, and Shendan Jin. Best-of-two-worlds analysis of online search. In *36th International Symposium on Theoretical Aspects of Computer Science, STACS 2019*, volume 126 of *LIPIcs*, pages 7:1–7:17, 2019.
- 5 Spyros Angelopoulos, Christoph Dürr, Shendan Jin, Shahin Kamali, and Marc P. Renault. Online computation with untrusted advice. In *11th Innovations in Theoretical Computer Science Conference, ITCS 2020*, pages 52:1–52:15, 2020.
- 6 Spyros Angelopoulos, Christoph Dürr, and Thomas Lidbetter. The expanding search ratio of a graph. *Discrete Applied Mathematics*, 260:51–65, 2019.

- 7 Spyros Angelopoulos and Shahin Kamali. Contract scheduling with predictions, 2020. Available as [arXiv:2011.12439](https://arxiv.org/abs/2011.12439).
- 8 Yossi Azar, Andrei Z. Broder, and Mark S. Manasse. On-line choice of on-line algorithms. In Vijaya Ramachandran, editor, *Proceedings of the Fourth Annual ACM/SIGACT-SIAM Symposium on Discrete Algorithms, 25-27 January 1993, Austin, Texas, USA*, pages 432–440. ACM/SIAM, 1993.
- 9 Ricardo A. Baeza-Yates, Joseph C. Culberson, and Gregory G.E. Rawlins. Searching in the plane. *Information and Computation*, 106:234–244, 1993.
- 10 Anatole Beck. On the linear search problem. *Naval Research Logistics*, 2:221–228, 1964.
- 11 Anatole Beck. More on the linear search problem. *Israel Journal of Mathematics*, 3(2):61–70, 1965.
- 12 Anatole Beck and Micah Beck. Son of the linear search problem. *Israel Journal of Mathematics*, 48(2-3):109–122, 1984.
- 13 Anatole Beck and Micah Beck. The linear search problem rides again. *Israel Journal of Mathematics*, 53(3):365–372, 1986.
- 14 Anatole Beck and Micah Beck. The revenge of the linear search problem. *SIAM journal on control and optimization*, 30(1):112–122, 1992.
- 15 Anatole Beck and D.J. Newman. Yet more on the linear search problem. *Israel Journal of Mathematics*, 8:419–429, 1970.
- 16 Anatole Beck and Peter Warren. The return of the linear search problem. *Israel Journal of Mathematics*, 14(2):169–183, 1973.
- 17 R. Bellman. An optimal search problem. *SIAM Review*, 5:274, 1963.
- 18 Daniel S. Bernstein, Lev Finkelstein, and Shlomo Zilberstein. Contract algorithms and robots on rays: unifying two scheduling problems. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1211–1217, 2003.
- 19 Daniel S. Bernstein, Theodore J. Perkins, Shlomo Zilberstein, and Lev Finkelstein. Scheduling contract algorithms on multiple processors. In *Proceedings of the Eighteenth AAAI Conference on Artificial Intelligence (AAAI)*, pages 702–706, 2002.
- 20 Prosenjit Bose, Jean-Lou De Carufel, and Stephane Durocher. Searching on a line: A complete characterization of the optimal solution. *Theoretical Computer Science*, 569:24–42, 2015.
- 21 Joan Boyar, Lene M. Favrholdt, Christian Kudahl, Kim S. Larsen, and Jesper W. Mikkelsen. Online algorithms with advice: A survey. *SIGACT News*, 47(3):93–129, 2016.
- 22 Jurek Czyzowicz, Konstantinos Georgiou, Evangelos Kranakis, Danny Krizanc, Lata Narayanan, Jaroslav Opatrny, and Sunil M. Shende. Search on a line by byzantine robots. In *27th International Symposium on Algorithms and Computation, ISAAC*, volume 64 of *LIPIcs*, pages 27:1–27:12. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016.
- 23 Erik D. Demaine, Sándor Fekete, and Shmuel Gal. Online searching with turn cost. *Theoretical Computer Science*, 361:342–355, 2006.
- 24 Rudolf Fleischer, Tom Kamphans, Rolf Klein, Elmar Langetepe, and Gerhard Trippen. Competitive online approximation of the optimal search ratio. *SIAM Journal on Computing*, 38(3):881–898, 2008.
- 25 Shmuel Gal. A general search game. *Israel Journal of Mathematics*, 12:32–45, 1972.
- 26 Shmuel Gal. Minimax solutions for linear search problems. *SIAM Journal on Applied Mathematics*, 27:17–30, 1974.
- 27 Patrick Jaillet and Matthew Stafford. Online searching. *Operations Research*, 49:234–244, 1993.
- 28 Ming-Yang Kao and Michael Littman. Algorithms for informed cows. In *Proceedings of the AAAI 1997 Workshop on Online Search*, 1997.
- 29 Ming-Yang Kao, Yuan Ma, Michael Sipser, and Yiqun Yin. Optimal constructions of hybrid algorithms. *Journal of Algorithms*, 29(1):142–164, 1998.

- 30 Ming-Yang Kao, John H Reif, and Stephen R Tate. Searching in an unknown environment: an optimal randomized algorithm for the cow-path problem. *Information and Computation*, 131(1):63–80, 1996.
- 31 David G. Kirkpatrick. Hyperbolic dovetailing. In *Proceedings of the 17th European Symposium on Algorithms (ESA)*, pages 616–627, 2009.
- 32 Elias Koutsoupias, Christos H. Papadimitriou, and Michalis Yannakakis. Searching a fixed graph. In *Proc. of the 23rd Int. Colloq. on Automata, Languages and Programming (ICALP)*, pages 280–289, 1996.
- 33 Andrey Kupavskii and Emo Welzl. Lower bounds for searching robots, some faulty. In *Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing*, pages 447–453, 2018.
- 34 Alejandro López-Ortiz and Sven Schuierer. The ultimate strategy to search on  $m$  rays? *Theoretical Computer Science*, 261(2):267–295, 2001.
- 35 Alejandro López-Ortiz and Sven Schuierer. On-line parallel heuristics, processor scheduling and robot searching under the competitive framework. *Theoretical Computer Science*, 310(1–3):527–537, 2004.
- 36 Thodoris Lykouris and Sergei Vassilvitskii. Competitive caching with machine learned advice. In *Proceedings of the 35th International Conference on Machine Learning, ICML*, pages 3302–3311, 2018.
- 37 Arya Mazumdar and Barna Saha. Clustering with noisy queries. In *Advances in Neural Information Processing Systems 30*, pages 5788–5799. Curran Associates, Inc., 2017.
- 38 Andrew McGregor, Krzysztof Onak, and Rina Panigrahy. The oil searching problem. In *Proceedings of the 17th European Symposium on Algorithms (ESA)*, pages 504–515, 2009.
- 39 Manish Purohit, Zoya Svitkina, and Ravi Kumar. Improving online algorithms via ML predictions. In *Advances in Neural Information Processing Systems*, volume 31, pages 9661–9670, 2018.
- 40 Sven Schuierer. Lower bounds in online geometric searching. *Computational Geometry: Theory and Applications*, 18(1):37–53, 2001.
- 41 Sven Schuierer. A lower bound for randomized searching on  $m$  rays. In *Computer Science in Perspective*, pages 264–277, 2003.



# Sequential Defaulting in Financial Networks

Pál András Papp

ETH Zürich, Switzerland  
apapp@ethz.ch

Roger Wattenhofer

ETH Zürich, Switzerland  
wattenhofer@ethz.ch

---

## Abstract

We consider financial networks, where banks are connected by contracts such as debts or credit default swaps. We study the clearing problem in these systems: we want to know which banks end up in a default, and what portion of their liabilities can these defaulting banks fulfill. We analyze these networks in a sequential model where banks announce their default one at a time, and the system evolves in a step-by-step manner.

We first consider the *reversible model* of these systems, where banks may return from a default. We show that the stabilization time in this model can heavily depend on the ordering of announcements. However, we also show that there are systems where for any choice of ordering, the process lasts for an exponential number of steps before an eventual stabilization. We also show that finding the ordering with the smallest (or largest) number of banks ending up in default is an NP-hard problem. Furthermore, we prove that defaulting early can be an advantageous strategy for banks in some cases, and in general, finding the best time for a default announcement is NP-hard. Finally, we discuss how changing some properties of this setting affects the stabilization time of the process, and then use these techniques to devise a *monotone model* of the systems, which ensures that every network stabilizes eventually.

**2012 ACM Subject Classification** Applied computing → Economics; Theory of computation → Market equilibria; Theory of computation → Network games

**Keywords and phrases** Financial Network, Sequential Defaulting, Credit Default Swap, Clearing Problem, Stabilization Time

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.52

**Related Version** The full version of the paper is available at <https://arxiv.org/abs/2011.10485>.

## 1 Introduction

The world's financial system is a highly complex network where banks and other financial institutions are interconnected by various kinds of contracts. These connections create a strong interdependence between the banks: if one of them goes bankrupt, then this also affects others, causing a cascading effect through the network. Such ripple effects also had an important role in the financial crisis of 2008, and hence there is an increasing interest in the network-based properties of these systems.

One fundamental question in these networks is the so-called *clearing problem*: given a network of banks and contracts, we need to decide which of the banks can fulfill their payment obligations, and which of the banks cannot, and thus have to report a default. This question is of high interest both for financial authorities and for the banks involved.

With two simple kinds of contracts, one can already build a financial network model that captures a wide range of phenomena in real-life financial systems. Previous work has mostly focused on the equilibrium states in these models, i.e. the fixed final states where the recovery rates of banks are consistent with their current assets and liabilities. However, in practice, most events in a financial system happen gradually, one after another: a single



© Pál András Papp and Roger Wattenhofer;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 52; pp. 52:1–52:20



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

bank announces a default, which might prompt another bank to reevaluate its situation, and also need to call being in default. This sequential development is an inherent part of the way financial networks behave, and as such, it is crucial to understand.

In particular, there is a range of natural questions that only arise if we study how the system develops in a step-by-step fashion. Can we reach every equilibrium state in a sequential manner? How does the ordering of default announcements influence the final outcome? Is there an optimal strategy of timing the announcements, either from a financial authority's or a single bank's perspective? How long can the sequential process last, and in particular, is it guaranteed to always stabilize eventually?

In this paper we analyze the development of financial systems in a sequential model, where banks update their situation one after another. We first study the *reversible model*, which is a natural sequential setting in such networks. We analyze this model from three main perspectives:

- *Stabilization time*: We show that a system can easily keep running infinitely in this model. Moreover, the time of stabilization heavily depends on the ordering of default announcements. We also present a more complex system that does stabilize eventually, but only after exponentially many steps.
- *Globally best solution*: We show that finding the ordering which results in the smallest (or largest) number of defaulting banks in the final state is NP-hard.
- *Defaulting strategies*: We study the best defaulting strategy of a single bank, and show that surprisingly, a bank may achieve the best outcome by announcing its default as early as possible. We also prove that in general, finding the best time to report a default is NP-hard.

Moreover, since the possibly infinite runtime is the most unrealistic aspect of this model, we analyze the reasons behind this phenomenon, and we discuss how it can be avoided in our sequential model.

- *Monotone sequential model*: We show that with two minor changes to the setting (a more sophisticated update rule and a slightly different handling of defaulting banks), we can develop a monotone model variant where the recovery rate of banks can only decrease, and the system is always guaranteed to stabilize after quadratically many steps. We also compare this setting to the reversible model in terms of defaulting strategies.

## 2 Related Work

The network-based analysis of financial systems has been rapidly gaining attention in the last decade. Most studies are based on the early financial network model of Eisenberg and Noe [11], which only assumes simple debt contracts between the banks. The propagation of shocks has been analyzed in many variants of this base model over the last decade [9, 5, 4, 1, 13, 15]; in particular, the model has been extended by default costs [20], cross-ownership relations [24, 12] or game-theoretic aspects [6].

However, the common ground in these model variants is that they can only describe *long positions* between the banks: a better outcome for one bank always means a better (or the same) outcome for other banks. This already allows us to capture how the default of a single bank can cause a ripple effect in the system, but it also ensures that there is always an equilibrium which is simultaneously best for all banks [11, 20]. As such, long positions cannot represent e.g. the opposing interests of banks in real-world financial systems. In particular, banks in practice often have *short positions* on each other when a worse situation for one bank is more favorable to another bank, mostly due to various kinds of financial derivatives.

The recent work of Schuldenzucker et. al. [22] presents a more refined model where the network also contains credit default swaps (CDSs) besides regular debt contracts. CDSs are financial derivatives that essentially allow banks to bet on the default of another bank in the system; they have played a dominant role in the financial crisis of 2008 [14], and have been thoroughly studied in the financial literature [10, 16]. While CDSs are still a rather simple kind of derivative, they already allow us to model short positions in the network; as such, their introduction to the system leads to remarkably richer behavior. In our paper, we also assume these two kinds of contracts in the network.

The work of [22, 23] discusses various properties of this new model: they show that systems may have multiple solutions (equilibrium states) in this model, and finding a solution is PPAD-complete. They also show that with default costs, these systems might not have a solution at all, and deciding whether a solution exists becomes NP-hard. The work of [19] studies a range of objective functions for selecting the best solution in this model, showing that the best equilibrium is not efficiently approximable to a  $n^c$  factor for any  $c < \frac{1}{4}$ . The work of [18] analyzes the model from a game-theoretical perspective, discussing how the removal or modification of contracts can lead to more favorable equilibria for the acting banks, and showing that such operations can lead to game-theoretical dilemmas.

However, all these results only analyze the model in terms of equilibrium states. This is indeed important when the market is hit by a large shock, and a central authority has to analyze the whole system, identify its equilibria, and possibly select one of them to artificially implement. However, apart from these rare occasions, the network mostly evolves sequentially, with banks announcing defaults in a step-by-step manner. For an understanding of real-world networks, it is also essential to study this gradually developing behavior of the process besides the equilibrial outcomes.

Sequential models of financial networks have already been studied in several papers; however, most of them consider some variant of the debt-only model with long positions [7, 2]. The paper of [22] notes that sequential clearing in their model would be dependent on the order of defaults, but does not investigate this direction any further. At the other end of the scale, the work of [3] introduces a very general sequential model (where payment obligations can be a function of all banks and all previous time steps), with a specific focus on expressing concrete real-world examples in this setting. As such, to our knowledge, there is no survey that considers a simple network model with both long and short positions, and analyzes the step-by-step development of financial systems in this model.

Finally, we point out that the clearing problem indeed has a high relevance in practice, e.g. when financial authorities conduct stress tests to analyze the sensitivity of real-world networks. One concrete example for a study of this problem is the European Central Bank's stress test framework [8].

## 3 Model Definition

### 3.1 Banks and contracts

Our financial system model consists of a set of *banks* (or *nodes*)  $B$ . We denote individual banks by  $u, v$  or  $w$ , and the number of banks by  $n = |B|$ . Banks are connected by two kinds of contracts that both describe a specific payment obligation from a debtor bank  $u$  to a creditor bank  $v$ . The amount of payment obligation is called the *weight* of the contract.

The simpler kind of connection is a *simple debt* contract, which obliges the debtor  $u$  to pay a specific amount  $\delta$  to the creditor  $v$ . This liability is unconditional, i.e.  $u$  owes this amount to  $v$  in any case.

Besides debts, banks can also enter into *conditional debt contracts* where the payment obligation depends on some external event in the system. One of the most frequent forms of such a conditional debt is a credit default swap (*CDS*), which obliges  $u$  to pay a specific amount to  $v$  in case a specific third bank  $w$  (the *reference entity*) is in default. More specifically, if  $w$  can only fulfill a  $r_w$  portion of its payment obligations (known as the recovery rate of  $w$ ), then a CDS of weight  $\delta$  implies a payment obligation of  $\delta \cdot (1 - r_w)$  from  $u$  to  $v$ . For simplicity, we assume that all conditional debt contracts are CDSs.

In practice, CDS contracts can, for example, be used by a bank as an insurance policy against the default of its debtors. If  $v$  suspects that its debtor  $w$  might not be able to fulfill its payment obligation, then  $v$  can enter into a CDS contract (as creditor) in reference to  $w$ ; if  $w$  goes into default and is indeed unable to pay, then  $v$  receives some payment on this CDS instead. However, banks may also enter into CDSs for other reasons, e.g. speculative bets about future developments in the market. As a sanity assumption, we assume that no bank can enter into a contract with itself or in reference to itself.

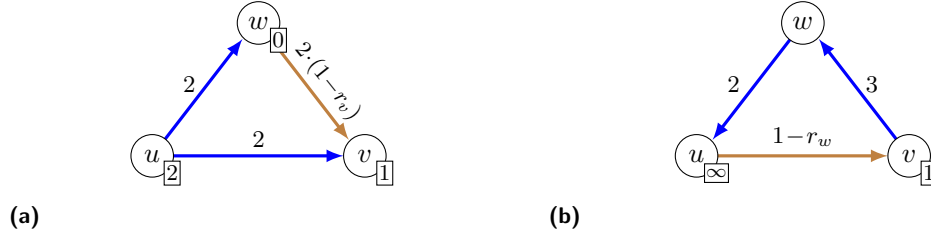
Besides the contracts between banks, a financial system is described by the amount of funds (in financial terms: *external assets*) owned by each bank, denoted by  $e_v$  for a specific bank  $v$ . The external assets and the incoming payments describe the total amount of assets available to  $v$ , while the outgoing contracts describe the total amount of payment obligations of  $v$ . If  $v$  is not able to fulfill all these obligations from its assets, then we say that  $v$  is *in default*. If  $v$  is in default, then the fraction of liabilities that  $v$  is able to pay is the *recovery rate* of  $v$ , denoted by  $r_v$ . Note that  $r_v \in [0, 1]$ , and  $v$  is in default if  $r_v < 1$ . We represent the recovery rates of all banks in a recovery rate vector  $r \in [0, 1]^B$ .

For an example, consider the financial system in Figure 1a with 3 banks. The banks have external assets of  $e_u = 2$ ,  $e_v = 1$  and  $e_w = 0$ . Bank  $u$  has a debt of weight 2 towards both  $v$  and  $w$ , and there is a CDS of weight 2 from  $w$  to  $v$ , with  $u$  as the reference entity. In this network,  $u$  has a total payment obligation of 4, but only has assets of 2, so  $u$  is in default, with a recovery rate of  $r_u = \frac{2}{4} = \frac{1}{2}$ . Bank  $u$  must use its funds of 2 to pay 1 unit of money to both  $w$  and  $v$ , proportionally to its obligations. Since  $r_u = \frac{1}{2}$ , the CDS from  $w$  to  $v$  will induce a payment obligation of  $2 \cdot (1 - r_u) = 1$ . The payment of 1 coming from  $u$  allows  $w$  to fulfill this obligation to  $v$ , thus narrowly avoiding default (hence  $r_w = 1$ ). Finally,  $v$  receives 1 unit from both  $u$  and  $w$ , has funds of 1 itself, and no payment obligations, so it has a positive equity of 3, and  $r_v = 1$ .

For convenience, we will use a simplified version of this notation in our figures: we only show the weight  $\delta$  of a contract when  $\delta \neq 1$ , and we only show the external assets of  $v$  explicitly if  $e_v \neq 0$ . We also write  $e_v = \infty$  to conveniently indicate that  $v$  can pay its liabilities in any case.

We also note that many of our constructions in the paper contain banks that have the exact same amount of assets and liabilities, like  $w$  in this example. This is a somewhat artificial “edge case” that still ensures  $r_w = 1$ . However, this is only for the sake of simplicity; we could avoid these edge cases by providing more assets to the banks in question.

Finally, we point out that contracts in a real-world financial system are often results of an earlier transaction between the banks, i.e. the creditor  $v$  previously offering a loan to the debtor  $u$ . We assume that such earlier payments are implicitly represented in  $e_u$ , and as such, the external assets and the contracts are together sufficient to describe the current state of the system.



■ **Figure 1** Two example systems on 3 banks. External assets are shown in rectangles besides the banks. Simple debts are denoted by blue arrows from debtor to creditor, while CDSs are denoted by light brown arrows from debtor to creditor, with the payment obligation shown beside the arrow.

### 3.2 Assets, liabilities and equilibria

We now formally define the liabilities and assets of banks in our systems. Note that due to the conditional debts, the payment obligations in a network are always a function of the recovery rate vector  $r$ .

Assuming a specific vector  $r$ , the liability  $l_{u,v}$  of a bank  $u$  towards a bank  $v$  is defined as the sum of payment obligation from  $u$  to  $v$  on all contracts, i.e.

$$l_{u,v}(r) = \delta_{u,v} + \sum_{w \in V} \delta_{u,v}^w \cdot (1 - r_w),$$

where  $\delta_{u,v}$  is the weight of the simple debt contract from  $u$  to  $v$  (if this contract exists, and 0 otherwise), and  $\delta_{u,v}^w$  is the weight of the CDS from  $u$  to  $v$  in reference to  $w$  (if it exists, and 0 otherwise). The total *liability* of  $u$  is simply the sum of liabilities to all other banks:  $l_u(r) = \sum_{v \in V} l_{u,v}(r)$ .

However, the actual *payment*  $p_{u,v}$  from  $u$  to  $v$  can be less than  $l_{u,v}$  if  $u$  is in default. If  $u$  is in default, then it has to spend all of its assets to make payments to creditors. Most financial system models assume that in this case,  $u$  has to follow the *principle of proportionality*, i.e. it has to make payments proportionally to the corresponding liabilities. This means that if  $u$  can pay an  $r_u$  portion of its total liabilities, and it has a liability of  $l_{u,v}$  towards  $v$ , then the payment from  $u$  to  $v$  is  $p_{u,v}(r) = r_u \cdot l_{u,v}(r)$ .

On the other hand, we can define the *assets* of a bank  $v$  as the sum of  $v$ 's external assets and its incoming payments in the network; that is,

$$a_v(r) = e_v + \sum_{u \in V} p_{u,v}(r).$$

If  $v$  is in default, then all these assets are used for  $v$ 's payment obligations; otherwise,  $a_v - l_v$  of these assets remain at  $v$ . Note that while both  $a_v(r)$  and  $l_v(r)$  are formally a function of  $r$ , we often simplify this notation to  $a_v$  and  $l_v$  when the recovery rate is clear from the context.

Recall that the recovery rate of  $v$  indicates the portion of payment obligations that  $v$  is able to fulfill. As such, a valid choice of  $r_v$  requires  $r_v = 1$  if we have  $a_v \geq l_v$ , and  $r_v = \frac{a_v}{l_v}$  if  $a_v < l_v$ . For simplicity, let us introduce a separate function  $R$  to denote this dependence on  $a_v$  and  $l_v$ ; that is, we define the function  $R : [0, \infty) \times [0, \infty) \rightarrow [0, 1]$  as

$$R(a, l) = \begin{cases} 1, & \text{if } a \geq l \\ \frac{a}{l}, & \text{otherwise.} \end{cases}$$

We say that a vector  $r \in [0, 1]^B$  is an *equilibrium* (or a *clearing vector*) of the system if for each bank  $v \in B$ , we have  $r_v = R(a_v(r), l_v(r))$ ; that is, if the recovery rate vector is consistent with the assets and liabilities it generates in the network. Previous work has

mostly focused on the analysis of different equilibrium states. Recall that while it is mostly straightforward to find the equilibrium states in our example constructions, the problem is PPA-hard in general [23].

We have already seen a simple example equilibrium in Figure 1a; for another example that is slightly more challenging to compute, let us consider Figure 1b. Here bank  $u$  is again always able to pay its liabilities, so  $r_u = 1$  in any case. Furthermore, neither  $r_v = 1$  nor  $r_w = 1$  can provide an equilibrium in this network, so both  $v$  and  $w$  must be in default in any solution. Thus any equilibrium must have

$$r_v = \frac{a_v}{l_v} = \frac{1 + 1 - r_w}{3} \quad \text{and} \quad r_w = \frac{a_w}{l_w} = \frac{3 \cdot r_v}{2}.$$

This implies that the only equilibrium is  $r_v = \frac{4}{9}$ ,  $r_w = \frac{2}{3}$ .

### 3.3 Sequential models of defaulting

We have defined the equilibria of the system as the states  $r$  that would fulfill the payment criteria if every bank were to simultaneously update its recovery rate to  $r$ . However, in practice, the announcement of defaults usually happens in a sequential manner, due to different sources of delay in the system: even if it is clear from  $a_v$  and  $l_v$  that a bank  $v$  is only able to fulfill a specific  $r_v$  portion of its liabilities, this might not be immediately known to the creditors of  $v$  (due to incomplete information), or the legal framework may first allow  $v$  to try to obtain further funds before officially having to announce its default. As such, the officially announced recovery rate  $r_v$  might not always equal  $R(a_v, l_v)$ , and  $v$  has to explicitly announce the changes in  $r_v$  in order to make other banks aware of this situation.

Hence in our sequential model, each step of the process will consist of a single bank announcing an *update* to its recovery rate. That is, given the assets  $a_v$  and liabilities  $l_v$  currently available to  $v$ , if the official recovery rate  $r_v$  does not equal  $R(a_v, l_v)$ , then bank  $v$  can (and eventually has to) announce a new official recovery rate of  $r_v := R(a_v, l_v)$ . Since this affects both the payments received by the debtors of  $v$  and the payment obligations on CDSs in reference to  $v$ , it can have various effects on the system, providing new assets and liabilities to some banks; as a result, these banks may also end up with a higher or lower asset/liability balance than their currently announced recovery rate, and thus they will also have to execute a new update at some point.

More formally, we consider discrete time steps  $t = 0, 1, 2, \dots$ . Each step consists of a single bank  $v$  announcing an update to  $r_v$ . That is, if  $v$  has assets  $a_v^{(t-1)}$  and liabilities  $l_v^{(t-1)}$ , but a recovery rate of  $r_v^{(t-1)} \neq R(a_v^{(t-1)}, l_v^{(t-1)})$  at time  $t - 1$ , then we say that  $v$  is *updatable* at time  $t - 1$ . In each time step  $t$ , we select a bank  $v$  that is updatable at time  $t - 1$ , and define the state of the system at time  $t$  by (i) setting  $r_v^{(t)} = R(a_v^{(t-1)}, l_v^{(t-1)})$  for the bank  $v$  that executes the update, (ii) setting  $r_u^{(t)} = r_u^{(t-1)}$  for every other bank  $u \neq v$ , and (iii) calculating  $a_u^{(t)}$  and  $l_u^{(t)}$  for all  $u \in B$  based on this new vector  $r^{(t)}$ .

We assume that initially, each bank  $v$  has  $r_v^{(0)} = 1$ , and we compute  $a_v^{(0)}$  and  $l_v^{(0)}$  accordingly. We say that the sequential process *stabilizes* in round  $t$  if there is no updatable bank in round  $t$ .

## 4 Basic Properties

We begin by discussing some fundamental properties of this sequential setting.

### 4.1 Reversibility and infinite cycling

One important property of the sequential model is that even if a bank  $v$  goes into default, it can easily return from this default later. That is, future updates in the system might increase the payment obligation on an incoming CDS of  $v$ , thus increasing  $a_v$  and possibly raising  $\frac{a_v}{l_v}$  above 1 again. This is in line with real-world financial systems, where returning from a default is also often possible if a bank acquires new assets. Due to this property, we also refer to this setting as the *reversible model*.

Note that in practice, defaulting banks are often given a limited amount of time to obtain new assets and thus reverse a default; however, our sequential setting does not define an explicit timing of defaults (only their order), so such rules are not straightforward to include in our model. Nonetheless, we point out that many of our example constructions also work if we assume that defaults are only reversible for a specific (constant) number of rounds.

Another important property is that in a cyclic network topology, our model can easily result in an infinite loop of updates. Consider the example in Figure 2, where the default of  $v$  indirectly provides new assets to  $v$ . Since  $r_v = 1$  initially,  $u$  must first update to  $r_u = 0$ , and as a result,  $v$  must update to  $r_v = 0$ . However, this leads to new liabilities in the network, providing assets to both  $u$  and (indirectly) to  $v$ , so  $u$  (and then  $v$ ) must update its rate back to  $r_u = r_v = 1$ . This returns the system to its initial state, where  $u$  (and  $v$ ) will continue by updating their recovery rates to 0 again.

If we keep repeating these few steps, then  $u$  and  $v$  alternate between  $r_u = r_v = 0$  and  $r_u = r_v = 1$  endlessly. Note that the system does have an equilibrium in  $r_u = r_v = \frac{1}{2}$ ; however, instead of converging to this state, the banks keep on periodically repeating the same few steps. The possibility of such behavior in a sequential setting has already been noted in [22] or [3] before. While this looping behavior is certainly undesired, it follows straightforwardly from the reversibility of defaults and the existence of cycles in the network topology. As such, these situation could also occur in real-world systems, requiring a financial authority to intervene and set the system artificially to its equilibrium.

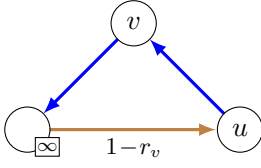
### 4.2 Dependence on the order of updates

Another key property of the sequential model is that the final outcome becomes dependent on the ordering of updates, i.e. whether some banks announce their default earlier or later.

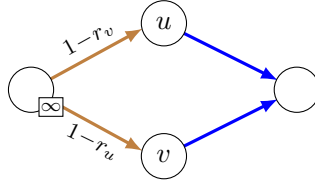
We show a simple example of this dependence on the *branching gadget* of Figure 3, which has already been used as a building block in the works of [23] and [19]. In this system, neither of the two banks  $u$  and  $v$  have any assets initially, so they are unable to fulfill their obligations. However, if  $u$  is the first one to report default (updating to a new recovery rate of  $a_v^{(0)}/l_v^{(0)} = 0$ ), then this provides 1 unit of new assets to  $v$ , which means that  $v$  does not default anymore; the system stabilizes with  $r_u = 0$ ,  $r_v = 1$ . Similarly, if  $v$  is the first one to execute an update, then this provides new assets to  $u$ , and the system stabilizes with  $r_u = 1$ ,  $r_v = 0$ . Thus both banks are strongly motivated to delay their default announcement as long as possible, as this might allow them to avoid defaulting entirely.

We can also note that there are further equilibrium states where both  $u$  and  $v$  are in default, e.g. when  $r_u = \frac{1}{2}$  and  $r_v = \frac{1}{2}$ ; due to its symmetry, one might even argue that this is the “fair” equilibrium to implement. However, this equilibrium is not reachable in any way through sequential updates; the only possible endstates of the sequential model are  $(r_u, r_v) = (0, 1)$  and  $(r_u, r_v) = (1, 0)$  as described above.

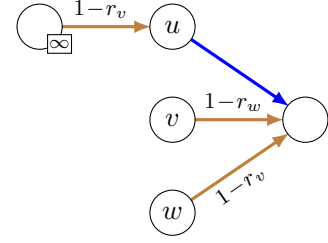




■ **Figure 2** Example system for an infinite loop in the reversible model.



■ **Figure 3** Example system where the outcome depends on the order of announcements.



■ **Figure 4** Example of an equilibrium that is not reachable in the sequential model.

This shows that even in terms of the final outcome, the sequential model can significantly differ from the static analysis of the system. This is not due to the presence of fractional recovery rates: we can also easily have equilibria with integer (i.e., 0 or 1) recovery rates that is not reachable in a sequential setting. In Figure 4, bank  $u$  is the only node who can execute an update, which immediately leads to the unique final state  $r_u = 0$ ,  $r_v = r_w = 1$ . However,  $r_u = 1$  with  $r_v = r_w = 0$  also forms an equilibrium in this system, so this phenomenon is indeed a result of the sequential nature of our model.

## 5 Results

We now move on to a deeper analysis of the model. We mainly focus on the length and outcome of the sequential process, and how the ordering of updates affects these properties.

Since our proofs will require more complex constructions, we switch to a simpler notation in our figures: instead of directly showing the liability  $\delta \cdot (1 - r_w)$  on a CDS, we only label the CDS by the weight  $\delta$  and the reference entity  $w$ , or simply by  $w$  when  $\delta = 1$ . Nonetheless, recall that each such CDS still denotes a liability of  $\delta \cdot (1 - r_w)$ .

### 5.1 Stabilization time

One fundamental question is the number of rounds it takes until the sequential process stabilizes, i.e. until no node can execute an update anymore. We first analyze this aspect in detail.

We have already seen in Figure 2 that even in simple examples, it can easily happen that the system does not stabilize at all.

► **Corollary 1.** *There is a system which never stabilizes.*

Furthermore, with the appropriate ordering of default announcements, we can also obtain any finite value as a stabilization time.

► **Lemma 2.** *For any integer  $k$ , there exists a system and an ordering such that the system stabilizes after exactly  $k$  steps.*

**Proof.** Consider the system on Figure 5. Similarly to Figure 2, this system allows us to produce an arbitrarily long sequence by switching only  $u$  and  $v$  repeatedly. However, when  $w$  announces a default, then both  $u$  and  $v$  gain enough assets to fulfill their obligations, so the system stabilizes after at most 2 more updates.

This allows us reach any magnitude of stabilization time, apart from a constant offset. We can then simply add  $O(1)$  more independent defaulting nodes to reach the desired value  $k$ . ◀

This already shows that stabilization time can heavily depend on the order of updates. A more extreme case of this is when the choice of the first update already decides between two very different outcomes for the system.

► **Lemma 3.** *There is a system where depending on the first update, the system either stabilizes in 1 step, or does not ever stabilize.*

**Proof.** Figure 6 is obtained by combining the base ideas of Figures 2 and 3. In this network, either bank  $w_1$  or  $w_2$  must execute the first update.

If  $w_2$  is the first to announce  $r_{w_2} = 0$ , then  $w_1$  receives a payment of 1, and the system immediately stabilizes; no other bank will make an update.

However, if we update  $r_{w_1} = 0$  first, then  $w_2$  survives, but on the other hand,  $u$  receives no assets at all. In this case, nodes  $u$  and  $v$  are in the same situation as in Figure 2, and thus the upper part of the system will never stabilize. ◀

Finally, infinite loops are not the only examples of long stabilization: it is also possible that the system does stabilize eventually, but for any ordering of updates, this only happens after exponentially many steps.

► **Theorem 4.** *There is a system where for any possible ordering, the system eventually stabilizes, but only after  $2^{\Omega(n)}$  steps.*

**Proof sketch.** This proof requires a significantly more complex construction than our previous statements. We only outline the main idea of the construction here, and we discuss the details in the full version of the paper.

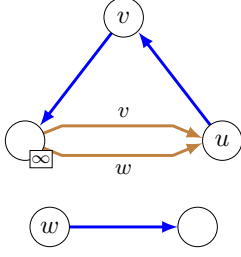
The first step of the proof is to build a *stable bit gadget*, which represents a mutable binary variable. The gadget offers a simple interface to set the bit to 0 or 1 through external conditions, and otherwise maintains its current value until the next such operation is executed.

Besides this, we create gadgets that describe logical states of an abstract process, similarly to a finite automaton. We also encode conditional transitions between these state gadgets, i.e. ensure that the system can only enter a given logical state if some banks currently have a specific recovery rate. This allows us to describe a logical process where the next state of the system is always determined by the current state and the current value of some stable bit gadgets.

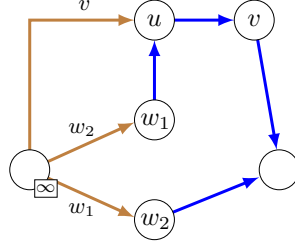
Using these tools, we can essentially design a binary counter on  $k = \Omega(n)$  bits, with  $k$  stable bits representing the bits of the counter. This counter will proceed to count from 0 to  $2^k - 1$ , and only stabilize after the counting has finished, resulting in a sequence of at least  $2^k$  steps.

The most challenging task is to ensure that in every step of the process, there is only one possible update we can execute next: the appropriate next step of the counting procedure. To achieve this, we not only need to ensure that some banks become updatable at specific times, but we also have to force the banks to indeed execute these updates, by encoding them as requirements in the transition conditions of our logical states. This results in a heavily restricted construction where there is essentially only one valid ordering of updates: the one that corresponds to the step-by-step incrementation of the binary counter. ◀

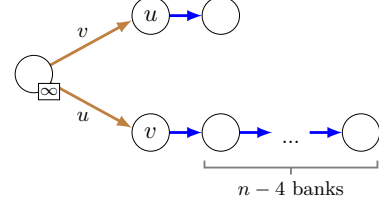
Note that another possible approach for measuring the stabilization time of our systems is to consider the number of *defaulting steps*, i.e. to only count the steps when a bank  $v$  updates from  $r_v = 1$  to  $r_v < 1$ . One can check that our results on stabilization time also hold for this alternative metric.



■ **Figure 5** Example system for Lemma 2. Recall that a CDS labeled with  $v$  still describes a payment obligation of  $1 - r_v$ .



■ **Figure 6** Example system for Lemma 3, where stabilization time depends on the choice of the first update.



■ **Figure 7** Example system for Lemma 6, i.e. where the number of defaults depends on the choice of the first update.

Finally, as a theoretical curiosity, we point out that our binary variable and state machine gadgets in the proof of Theorem 4 demonstrate that we can essentially use financial networks as a model of computation. We discuss the expressive power of this model in the full version of the paper.

► **Theorem 5.** *We can use financial networks to simulate any Turing-machine with a finite tape.*

## 5.2 Outcome with the fewest defaults

In case of a larger shock, a financial authority could also be interested in the final state of the system, and in particular, the number of banks that end up in default. This can again heavily depend on the order of updates; in fact, even a single decision in the ordering can be critical from this perspective.

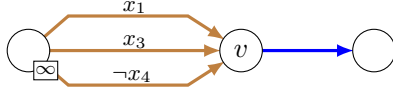
► **Lemma 6.** *Depending on the first update, the number of defaults can be either  $O(1)$  or  $n - O(1)$ .*

**Proof.** Consider the system on Figure 7. If  $u$  is the first to report a default with  $r_u = 0$ , then  $v$  receives 1 unit of payment, and thus no other node defaults. On the other hand, if  $v$  reports a default first, then  $u$  survives, but all the nodes in the lower chain have no incoming assets, and thus they all have to report a default eventually. So based on the first update, the number of defaults is either 1 or  $n - 3$ . ◀

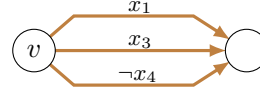
Hence if the authority has some influence over the ordering of updates, e.g. by allowing more flexibility to some banks than to others, then it could dramatically reduce the number of banks that end up in default. Unfortunately, even if we have complete control over the ordering, it is still hard to find the best possible ordering (in terms of the number of defaults in the final outcome).

► **Theorem 7.** *It is NP-hard to find the number of defaulting nodes in the best possible ordering.*

**Proof.** We reduce the question to the MAXSAT problem: given a boolean formula in conjunctive normal form, the goal of MAXSAT is to find the assignment of variables that satisfies the highest possible number of clauses [17].



■ **Figure 8** Clause gadget for the MAXSAT reduction in Theorem 7.



■ **Figure 9** Clause gadget for the MAXSAT reduction in Theorem 8.

Assume we have a MAXSAT problem on  $k$  variables  $x_1, \dots, x_k$ , and  $m$  clauses. Note that in our financial systems, the branching gadget of Figure 3 is a natural candidate for representing a boolean variable, since in any sequence, exactly one of  $u$  and  $v$  will eventually default. We point out that this gadget has already been used for similar purposes before in [23] and [19].

Hence for each variable  $x_i$ , we create a separate branching gadget in our system, and consider node  $u$  to represent the literal  $x_i$ , and node  $v$  to represent the literal  $\neg x_i$ . That is, we will consider  $x_i = \text{TRUE}$  if  $u$  defaults, while we consider  $x_i = \text{FALSE}$  if  $v$  defaults.

Furthermore, for each clause of the input formula, we create the clause gadget shown in Figure 8, with the CDSs labeled by the banks representing the literals in the clause. For example, the gadget in the figure is obtained for the clause  $(x_1 \vee x_3 \vee \neg x_4)$ . If any of the banks  $x_1$ ,  $x_3$  or  $\neg x_4$  default, then  $v$  receives enough assets to pay its debt, whereas otherwise,  $v$  must eventually default.

If we aim to avoid as many defaults as possible, then the reasonable ordering strategy is to first evaluate all the variable gadgets, and the clause gadgets only afterwards. In this case, each bank  $v$  of a clause gadget survives if and only if there is a true literal in the corresponding clause. This way the number of defaulting nodes in the final state is always exactly  $k$  in the variable gadgets, and at most  $m - \text{OPT}$  in the clause gadgets, where  $\text{OPT}$  denotes the maximal number of satisfiable clauses in our MAXSAT problem. Thus the minimal number of defaulting nodes in the system is altogether  $k + m - \text{OPT}$ . Finding this value also allows us to determine  $\text{OPT}$ , which completes our reduction. ◀

To analyze the effects of a shock, one might also be interested in the worst possible ordering; a similar reduction shows that this is also hard to find.

► **Theorem 8.** *It is NP-hard to find the number of defaulting nodes in the worst possible ordering.*

**Proof.** We can apply the same reduction from MAXSAT as before; we only need to slightly change the clause gadgets. Consider the clause gadget of Figure 9 for the example clause  $(x_1 \vee x_3 \vee \neg x_4)$ . To maximize the number of defaulting banks in this system, we can first evaluate the variables gadgets, which then allows us to produce an extra default for each clause that has a true literal. Thus the maximum number of defaulting nodes is  $k + \text{OPT}$ , which completes our reduction. ◀

### 5.3 Individual defaulting strategies

It is also natural to consider the effect of the ordering from the perspective of a single bank  $v$ . More specifically, is  $v$  motivated to immediately report its own default? Can it achieve a better outcome for itself by carefully timing its updates?

Intuitively, one would expect that banks are motivated to report their default as late as possible, in hope of obtaining further assets in the meantime. This is indeed true in many cases. For example, in the branching gadget of Figure 3,  $u$  and  $v$  clearly have a short position

in each other, and if either of them can wait long enough such that the other bank reports a default first, then it obtains new assets from the incoming CDS and thus manages to avoid a default entirely.

However, due to the complex interconnections in a network, it is in fact also possible that  $v$  achieves a better outcome if it reports a default earlier; it is even possible that this is the only strategy which allows  $v$  to avoid a default in the endstate of the system. We consider this one of our most surprising results.

► **Theorem 9.** *There exists a system where a bank  $v_1$  can only avoid a default in the final state of the system if  $v_1$  is the first bank to report a default.*

**Proof.** Consider the system in Figure 10, where only  $v_1$  or  $v_2$  can report a default initially, since no other node has any liabilities.

Assume that  $v_1$  is the first to report a default, updating to  $r_{v_1} = 0$ . This influences the network in two ways:  $v_2$  obtains assets of 1, and  $u_2$  now has a new liability of 1 as a result.

Thus the next update can only be executed by  $u_2$ , resulting in  $r_{u_2} = 0$ . On the one hand, this provides assets to  $u_1$ ; on the other hand, it creates liabilities for  $w_2$ . As a result, the next update can only be executed by  $w_2$ .

When  $w_2$  announces  $r_{w_2} = 0$ , this results in more liabilities for the defaulting  $u_2$ , and more assets for  $v_1$ . These assets make  $v_1$  the only updatable next node, bringing  $v_1$  back from its default with  $r_{v_1} = 1$ .

When  $v_1$  announces  $r_{v_1} = 1$ , then  $u_2$  loses some of its liabilities, and  $v_2$  loses its assets. This does not affect  $u_2$ , which remains at  $r_{u_2} = 0$  due to the default of  $w_2$ ; however,  $v_2$  now also has to report a default. The system finally stabilizes after  $v_2$  updates to  $r_{v_2} = 0$ : the assets/liabilities of  $v_1$  and  $u_1$  are affected, but neither of them has to make an update. Thus the final solution has  $r_{v_1} = 1$  and  $r_{v_2} = 0$ .

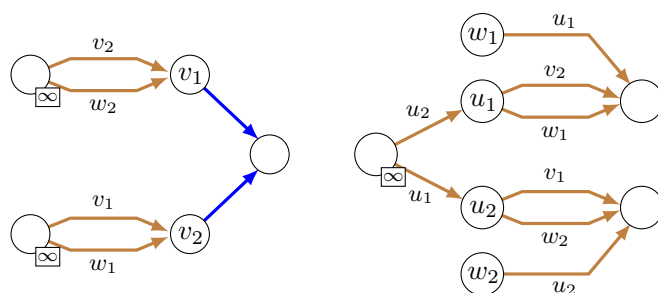
On the other hand, if  $v_2$  is the first to report default, then due to the symmetry of the system, the final outcome will have  $r_{v_1} = 0$  and  $r_{v_2} = 1$ . Note that in both cases, after the first update is executed, the remaining steps are already determined, and no alternative ordering is possible. Hence the only way for  $v_1$  to avoid a default in the final outcome is to be the first one to report a default. ◀

We can also show that in general, it is NP-hard to find the best default-reporting strategy for a bank. This even holds if the behavior of the rest of the network is “predictable”, i.e. if there is essentially only one ordering that the system can follow. This implies that any interpretation of this problem, e.g. optimizing a bank’s best-case payoff or worst-case payoff, is also hard.

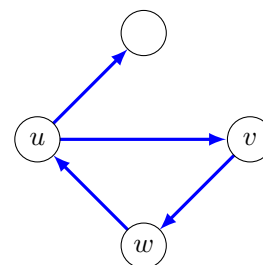
► **Theorem 10.** *It is NP-hard to find the time of defaulting that provides the highest payoff to a specific bank in the final outcome.*

**Proof sketch.** The main idea of the proof is to combine the binary counter construction of Theorem 4 with the MAXSAT reduction. That is, given a binary counter on  $k = \Theta(n)$  bits, we add a new node  $v$  to the system such that

- (a)  $v$  can choose to default anytime,
- (b) the default of  $v$  terminates the counting process, stabilizing the counter in its current state,
- (c)  $v$  then comes back from its default, and its assets in the final state are proportional to the amount of clauses satisfied in a SAT formula, where the value of the variables is derived from the finalized state of the bits in the counter.



■ **Figure 10** Example where early defaulting is the best strategy, with multiple source and sink nodes for a cleaner topology.



■ **Figure 11** Infinite convergence to an equilibrium state.

This means that the counter essentially enumerates all the possible value assignments of the variables, and the best defaulting strategy is obtained if counting is terminated at the assignment that satisfies the highest number of clauses. However, finding this assignment is NP-hard.

The details of the construction are discussed in the full version of the paper. ◀

## 6 Achieving Stabilization

While the reversible sequential model is realistic from many perspectives, the infinite looping property is clearly not reasonable in real-world systems. As such, it is natural to ask if there is a way to modify the model to avoid this situation, and instead ensure that every financial system stabilizes eventually.

In this section, we investigate the causes of this infinite behavior in the sequential model. We first show that we require more sophisticated update rules to avoid a specific kind of infinite behavior, namely when the system converges to an equilibrium. We then discuss liability freezing, a different (but in some sense also realistic) approach of handling defaulting banks in the network. Finally, we show that if we combine these two modifications, we can obtain a monotone sequential model where our systems always stabilize after polynomially many steps.

### 6.1 More sophisticated update rules

**Convergence to an equilibrium.** Since the addition of conditional debt contracts drastically increases the complexity of the model, it is a natural first assumption that such an infinite pattern can only arise if the system contains a CDS. However, this is not the case: we can also obtain a (slightly different kind of) infinite sequence in systems with only regular debts.

Consider the example system in Figure 11. Since bank  $u$  has  $l_u = 2$  and  $a_u = 1$  initially, it can begin by updating its recovery rate to  $r_u = \frac{1}{2}$ . As a result,  $v$  and  $w$  must also announce recovery rates of  $r_v = r_w = \frac{1}{2}$ . With  $a_u = \frac{1}{2}$ , bank  $u$  now has to update to  $r_u = \frac{1}{4}$ , which then gives  $r_v = r_w = \frac{1}{4}$ . Each such round prompts another round of updates, slowly converging to  $r_u = r_v = r_w = 0$ . While this is indeed the only equilibrium of the system, the process takes infinitely many steps to reach this state.

**Explicit computation of equilibria.** Such a convergence process can easily occur in any network with cycles; as real-world financial systems are also known to contain cycles [21], we can easily encounter such a situation in practice. In this case, it seems that a financial authority (or the banks involved) have no other option than to explicitly compute this equilibrium, and set their recovery rates to the appropriate values.

Fortunately, it is known that in case of fixed liabilities in the network (i.e. only simple debts), this is computationally feasible: there always exists a single maximal solution that is simultaneously best for all banks, and this solution can be found in polynomial time [20], essentially by repeatedly solving a system of linear equations. Thus an authority could indeed find this solution, and banks could directly update to these recovery rates in order to skip the convergence steps.

This allows us to introduce the notion of *smart updates*: after each updating step, we can consider the current liabilities in the network fixed, and we assume that the equilibrium of the system is computed under these liabilities (essentially reducing the convergence process to a single step). This equilibrium defines a *tentative recovery rate* for each bank  $v$ , denoted by  $\bar{r}_v$ . In smart updates, we assume that whenever  $v$  executes an update, it always updates to  $r_v := \bar{r}_v$ .

In the example of Figure 11, this means that the tentative recovery rates  $\bar{r}_u = \bar{r}_v = \bar{r}_w = 0$  are already computed initially, and thus any bank executing an update will immediately set its recovery rate to 0. This way the process already stabilizes after each bank has executed one update. In general, we achieve stabilization in this setting when  $r_v = \bar{r}_v$  for each bank  $v$  in the network.

While the explicit computation of equilibria may seem artificial, in practice, defaulting banks are often subject to more thorough supervision by the authorities. As such, it is not so unrealistic that the situation of a defaulting bank  $v$  is first analyzed by an authority, and this analysis determines the official recovery rate of  $v$ .

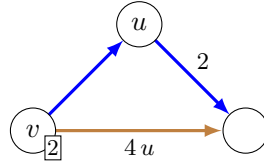
Also, recall that while equilibria are easy to find in debt-only networks, the introduction of CDSs changes this picture entirely. With CDSs, there can easily be multiple equilibria that are Pareto-optimal, and finding any of them is already a PPAD-hard problem [23]. Thus this explicit computation of  $\bar{r}_v$  is only possible for a single step of the process, when we consider the current payment obligation on each CDS fixed. As defaults rarely happen simultaneously in practice, it can indeed be realistic to assume that we can analyze the current (fixed) liabilities in the network after each new update.

Finally, note that smart updating is not yet enough to avoid an infinite convergence. In the system shown in Figure 12,  $v$  can initially fulfill its obligations, while  $u$  must update to  $r_u = \frac{1}{2}$ . This creates new liabilities of 2 for  $v$ , leading to the tentative recovery rates  $\bar{r}_v = \frac{2}{3}$  and thus  $\bar{r}_u = \frac{1}{3}$  after this first step. If  $u$  updates again (to  $r_u = \frac{1}{3}$ ), then the liability on the CDS again increases, and thus the next computed equilibrium has an even lower  $\bar{r}_u$ .

Each step of this process provides new tentative recovery rates, obtained as  $\bar{r}_v = \frac{2}{5-4 \cdot r_u}$  and  $\bar{r}_u = \frac{\bar{r}_v}{2} = \frac{1}{5-4 \cdot r_u}$ . This results in an infinite convergence to the equilibrium  $r_v = \frac{1}{2}$ ,  $r_u = \frac{1}{4}$ . Note that we can observe this behavior regardless of whether  $v$  ever updates its recovery rate to the new  $\bar{r}_v$  value; the assets of  $u$  are calculated independently of the recovery rate reported by  $v$ .

**Optimistic updates.** Another natural variant of smart updates is the *optimistic update* rule. To avoid the convergence phenomenon of Figure 11, this setting also assumes that the system is analyzed by an authority after each update. However, defaulting and non-defaulting nodes are now handled in a different manner in this analysis. More specifically, if a bank  $v$  is not





■ **Figure 12** Example of infinite convergence in a network, even in case of smart updates. Recall that the label  $4u$  on the CDS describes a payment obligation of  $4 \cdot (1 - r_u)$ .

in default (it has  $r_v = 1$  currently), then it is given the benefit of a doubt: we assume that it can fulfill its obligations, regardless of how many assets it currently has. On the other hand, banks in default are handled the same way as in case of smart updates.

This distinction can indeed be realistic: if  $v$  is non-defaulting, then  $a_v$  might not even be known to other banks, so the creditors of  $v$  have no better option than to assume that they will receive all payments from  $v$ . On the other hand, the assets of defaulting banks are under more thorough scrutiny in most legal frameworks.

Formally, optimistic update means that after each step of the process, we use a modified version of the liability network to compute the equilibrium. Whenever there is a contract of current weight  $\delta$  from  $u$  to  $v$  with  $r_u = 1$ , then we remove this contract from the network, and instead (i) we add a new debt of weight  $\delta$  from  $u$  to an artificial sink node  $s$ , ensuring that  $u$  still has this liability, and (ii) we increase the value of  $e_v$  by  $\delta$ , ensuring that  $v$  always has these assets. In contrast, if  $r_u < 1$ , we do not execute any changes on the outgoing contracts. This modified network ensures that until a bank reports a default, its lack of assets does not affect its creditors. We then use the same algorithm of [20] to find the equilibrium in this modified system, and set the next tentative recovery rates accordingly.

Revisiting the system in Figure 12, we see that bank  $u$  can again first update to  $r_u = \frac{1}{2}$ , which results in  $\bar{r}_v = \frac{2}{3}$ . However, with optimistic updates,  $u$  cannot make an update again: until  $v$  adjusts its recovery rate to this new value, the tentative recovery rate of  $u$  remains  $\frac{1}{2}$ , since we still expect to get the entire payment from the non-defaulting  $v$ . Note, however, that optimistic updating still does not prevent an infinite convergence in this system if, for example,  $u$  and  $v$  keep on updating alternately.

## 6.2 Liability freezing

We have seen that neither smart nor optimistic updating prevents an infinite sequential process in itself. For this, we also need to change another aspect of our model, namely how the contracts of  $v$  are handled once  $v$  goes into default.

Debts are rather simple from this perspective: they describe a previously established payment obligation in the network, so there is no incentive to change them if  $v$  defaults.

CDSs, however, pose a more complicated question, since they describe payment obligations that are dynamically changing. So far, we assumed that even after  $v$  defaults, the payment obligations on its CDSs keep changing as the reference entities are updated. Another possible approach is to assume *liability freezing*: whenever  $v$  goes into default, the liabilities on any incoming or outgoing CDS are fixed at the current value for the rest of the process. That is, a CDS with weight  $\delta$  and reference entity  $w$  at time  $t$  is essentially converted into a simple debt contract with weight  $\delta \cdot (1 - r_w^{(t)})$ , and this weight does not change in the future, even if  $r_w$  is updated.

This can be realistic when there is a larger time difference between subsequent defaults: by the time the next default happens, the previous bank has already completed the first phase of the insolvency process, and its incoming/outgoing payments have been established

and fixed. Indirectly, such a framework suggests that if  $v$  defaults, then it is expected to immediately “cash in” its incoming debts and fulfill its payment obligations, and not wait for a more favorable situation.

The main advantage of this approach is that if we combine liability freezing with optimistic updates, it provides a *monotone sequential model* where recovery rates can only decrease throughout the process. Intuitively, when a bank  $w$  makes an update, then CDSs in reference to  $w$  could only provide more assets to a bank  $v$  if we still have  $r_v = 1$ , as otherwise the liability on the CDS is already fixed. However, if  $r_v = 1$ , then the optimistic approach assumes anyway that  $v$  can pay its liabilities, and thus the update has no effect on other banks in the system.

This monotonic property ensures that any system stabilizes eventually in this model; on the other hand, it also means that once a bank  $v$  announces a default in this model, it has no possibility to reverse this default in the future, and its recovery rate can only get smaller with further updates.

By revisiting Figure 2, we can observe that both liability freezing and optimistic updates are crucial ingredients to achieve this monotonicity. Without liability freezing, the system loops infinitely if  $u$  and  $v$  make updates in an alternating fashion, both with smart and with optimistic updates. On the other hand, if we combine liability freezing with smart updates, then  $v$  can still alternate between  $r_v = 0$  and  $r_v = 1$  indefinitely; if  $u$  never makes an update, then the liability on the CDS will never be fixed at a specific value.

### 6.3 Stabilization in the monotone model

We now discuss the main properties of the monotone model. We first show that the model indeed ensures an eventual stabilization for any ordering. The key observation for this is that the recovery rate of banks can never increase in this model.

► **Theorem 11.** *The recovery rate of a bank can only decrease in the monotone model.*

**Proof.** The main idea is to show that for any bank  $v$ ,  $\bar{r}_v$  can only increase if we still have  $r_v = 1$  currently. This shows that we can never have  $\bar{r}_v > r_v$ , and thus no update can increase  $r_v$ .

Assume that node  $w$  updates  $r_w$  in a specific step, and assume for contradiction that this is the first step that increases  $\bar{r}_v$  for some bank  $v$  with  $r_v < 1$ . This means that the current update is still a decrease of  $r_w$ , since we must have  $\bar{r}_w < r_w$ . The update of  $r_w$  can have two kinds of effects on the system: it can change the liabilities on CDSs that are in reference to  $w$ , and it can result in a lower amount of assets for the creditors of  $w$ . We analyze these two effects separately.

Since the monotone model has liability freezing, the liability on a CDS from  $u$  to  $v$  (in reference to  $w$ ) can only change if we currently still have  $r_u = r_v = 1$ . Thus while this extra payment may increase  $\bar{r}_v$ , we will still have  $\bar{r}_v \leq r_v$  afterwards. Since the model uses optimistic updates and  $r_u = r_v = 1$ , both  $u$  and  $v$  only have debts towards the artificial sink  $s$  in the input graph of the equilibrium algorithm (which computes the tentative recovery rates), so the changes to  $\bar{r}_u$  and  $\bar{r}_v$  do not affect the tentative recovery rate of any other node.

As for the creditors of  $w$ , we consider two cases. If this is not a defaulting step (we already had  $r_w < 1$  before the update), then updating  $r_w$  does not change the liabilities in the input graph of the equilibrium algorithm (apart from the case of some non-defaulting nodes, as discussed above), so the tentative recovery rates will remain unchanged.

On the other hand, if this is a defaulting step, then the outgoing debts of  $w$  will now be redirected from  $s$  to the actual creditors of  $w$ . However, this operation can only result in less assets for a bank. More specifically, one can observe that any configuration of payments in this new graph is also a valid configuration of payments in the original graph before the redirection step. Hence if the  $\bar{r}_v$  value of any bank  $v$  increases with this step, then this contradicts the fact that the previous  $\bar{r}_v$  was obtained from a maximal equilibrium of the system. ◀

► **Theorem 12.** *The monotone model allows at most  $n$  defaulting and  $O(n^2)$  updating steps.*

**Proof.** Since recovery rates are always decreasing, every bank can default at most once, thus the number of defaulting steps is at most  $n$ .

For the  $O(n^2)$  upper bound, we show that there are at most  $n$  updating steps between any two consecutive defaulting steps. This is rather straightforward: recall from the proof of Theorem 11 that if bank  $w$  executes a non-defaulting update, then this can only change the value of  $\bar{r}_v$  for banks  $v$  that are not in default. Thus for any bank  $v$  in default,  $\bar{r}_v$  can not change between two defaulting steps of the process. This means that any bank can execute at most 1 updating step between two consecutive defaulting steps, limiting the number of steps in this period to  $n$ . ◀

We point out that this upper bound is asymptotically tight: we can easily construct a system and an ordering that indeed takes  $\Omega(n^2)$  steps in the monotone model. The construction does not even require CDSs in the network; it only contains simple debt contracts.

► **Lemma 13.** *There is a system with an ordering that lasts for  $\Omega(n)$  defaulting and  $\Omega(n^2)$  updating steps.*

**Proof.** Let  $m$  be a parameter with  $m = \Theta(n)$ , and consider Figure 13. All the banks  $w_1, \dots, w_m$  will eventually report a default in this system, so the number of defaulting steps is indeed  $m = \Omega(n)$ .

Let  $w_1, \dots, w_m$  report a default in this order throughout the process. After  $w_i$  has reported a default, bank  $v$  can always decrease its recovery rate to a new value of  $r_v = \frac{m-i}{m}$ . Finally, after each such update of  $v$ , assume that all the nodes  $u_1, \dots, u_m$  make an update step, also announcing a new recovery rate of  $\frac{m-i}{m}$ ; they can indeed all do this due to the update executed by  $v$ . This ordering has  $\Omega(m^2) = \Omega(n^2)$  updating steps altogether. ◀

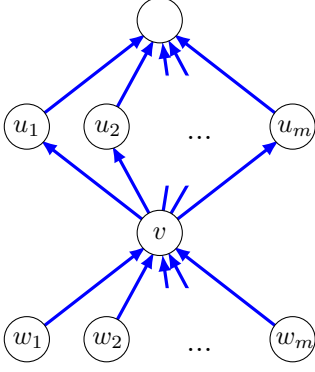
## 6.4 Defaulting strategies

Finally, we discuss how the monotone model compares to the reversible model in terms of defaulting strategies.

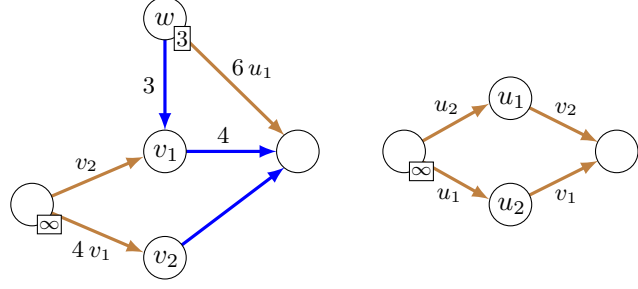
When finding the globally best ordering, the two models turn out to be very similar. In fact, our proofs from Section 5.2 can also be carried over to the monotone model without any changes.

► **Corollary 14.** *Lemma 6 and Theorems 7 and 8 also hold in the monotone model.*

In terms of individual defaulting strategies, the branching gadget again provides a simple example where late defaulting is beneficial: by delaying their updates, banks  $u$  and  $v$  can again entirely avoid a default.



■ **Figure 13** Example system for  $\Theta(n^2)$  stabilization time in the monotone model.



■ **Figure 14** Example system where early defaulting is the best strategy in the monotone model.

However, early defaulting is a more difficult question in this setting. In particular, we cannot hope for a result that is analogous to Theorem 9, since once a bank reports a default, there is no way to reverse this in the future. Nonetheless, early defaulting can still be a beneficial strategy in the monotone model: there are cases when a bank cannot avoid an eventual default in any way, but early defaulting can still allow the bank to have a higher recovery rate in the final state.

► **Theorem 15.** *There exists a system where a bank  $v$  only obtains its highest possible recovery rate in the final state of the system if  $v$  is the first bank to report a default.*

**Proof.** Consider the system in Figure 14, where either  $v_1$  or  $v_2$  can execute the first update. We analyze the defaulting strategies of bank  $v_1$  in this system.

Assume that  $v_1$  is the first to execute a step, announcing  $r_{v_1} = \frac{3}{4}$ . This gives new assets to  $v_2$  (resulting in  $\bar{r}_{v_2} = 1$ ), and new liabilities to  $u_2$  (resulting in  $\bar{r}_{u_2} = 0$ ). The next update can only be executed by  $u_2$ , setting  $r_{u_2} = 0$ ; at this point, the system stabilizes.

On the other hand, assume that  $v_2$  first announces  $r_{v_2} = 0$ . This provides  $\bar{r}_{v_1} = 1$  and  $\bar{r}_{u_1} = 0$ , so as a next step,  $u_1$  will announce a default. However, this results in new liabilities for  $w$ , so as a next step,  $w$  has to update to  $r_w = \frac{1}{3}$ . With this,  $v_1$  only has 2 assets altogether, so  $v_1$  must announce  $r_{v_1} = \frac{1}{2}$ . Hence  $v_1$  achieves a lower recovery rate in the final state if it is not the first bank to announce a default.

Note that with some further modifications, we can also make the example symmetric to ensure that both  $v_1$  and  $v_2$  are motivated to be the first one to default. ◀

Finally, one might also wonder if the monotone model allows an analogous result to Theorem 10, i.e. a hardness result on finding the best defaulting strategy of a single bank. However, note that the simple formulation of Theorem 10 was possible due to the fact that the proof construction only allowed one possible ordering in the rest of the system.

If we were to introduce a similar setting in the monotone model, then the banks could always find the best outcome in polynomial time, since the sequence can only last for  $O(n^2)$  steps. As such, in the monotone model, we can only expect similar hardness results for more complex formulations of this problem, such as finding the best defaulting time with respect to, e.g., the best-case or worst-case ordering of the remaining banks in the system.

## References

- 1 Daron Acemoglu, Vasco M Carvalho, Asuman Ozdaglar, and Alireza Tahbaz-Salehi. The network origins of aggregate fluctuations. *Econometrica*, 80(5):1977–2016, 2012.
- 2 Daron Acemoglu, Asuman Ozdaglar, and Alireza Tahbaz-Salehi. Systemic risk and stability in financial networks. *American Economic Review*, 105(2):564–608, 2015.
- 3 Tathagata Banerjee and Zachary Feinstein. Impact of contingent payments on systemic risk in financial networks. *Mathematics and Financial Economics*, 13(4):617–636, 2019.
- 4 Marco Bardoscia, Stefano Battiston, Fabio Caccioli, and Guido Caldarelli. Pathways towards instability in financial networks. *Nature Communications*, 8:14416, 2017.
- 5 Stefano Battiston, Guido Caldarelli, Robert M May, Tarik Roukny, and Joseph E Stiglitz. The price of complexity in financial networks. *Proceedings of the National Academy of Sciences*, 113(36):10031–10036, 2016.
- 6 Nils Bertschinger, Martin Hoefer, and Daniel Schmand. Strategic Payments in Financial Networks. In *11th Innovations in Theoretical Computer Science Conference (ITCS 2020)*, volume 151 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 46:1–46:16, Dagstuhl, Germany, 2020. Schloss Dagstuhl–Leibniz-Zentrum für Informatik.
- 7 Péter Csóka and P Jean-Jacques Herings. Decentralized clearing in financial networks. *Management Science*, 64(10):4681–4699, 2017.
- 8 Stéphane Dees, Jérôme Henry, and Reiner Martin. Stamp€: stress-test analytics for macro-prudential purposes in the euro area. *Frankfurt am Main: ECB*, 2017.
- 9 Gabrielle Demange. Contagion in financial networks: a threat index. *Management Science*, 64(2):955–970, 2016.
- 10 Darrell Duffie and Haoxiang Zhu. Does a central clearing counterparty reduce counterparty risk? *The Review of Asset Pricing Studies*, 1(1):74–95, 2011.
- 11 Larry Eisenberg and Thomas H Noe. Systemic risk in financial systems. *Management Science*, 47(2):236–249, 2001.
- 12 Matthew Elliott, Benjamin Golub, and Matthew O Jackson. Financial networks and contagion. *American Economic Review*, 104(10):3115–53, 2014.
- 13 Helmut Elsinger, Alfred Lehar, and Martin Summer. Risk assessment for banking systems. *Management science*, 52(9):1301–1314, 2006.
- 14 Ingo Fender and Jacob Gyntelberg. Overview: global financial crisis spurs unprecedented policy actions. *BIS Quarterly Review*, 13(4):1–24, 2008.
- 15 Matt V Leduc, Sebastian Poledna, and Stefan Thurner. Systemic risk management in financial networks with credit default swaps. *Available at SSRN 2713200*, 2017.
- 16 Yee Cheng Loon and Zhaodong Ken Zhong. The impact of central clearing on counterparty risk, liquidity, and trading: Evidence from the credit default swap market. *Journal of Financial Economics*, 112(1):91–115, 2014.
- 17 Christos H Papadimitriou. *Computational complexity*. Addison-Wesley, 1994.
- 18 Pál András Papp and Roger Wattenhofer. Network-Aware Strategies in Financial Systems. In *47th International Colloquium on Automata, Languages, and Programming (ICALP 2020)*, volume 168 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 91:1–91:17, Dagstuhl, Germany, 2020. Schloss Dagstuhl–Leibniz-Zentrum für Informatik.
- 19 Pál András Papp and Roger Wattenhofer. Default ambiguity: Finding the best solution to the clearing problem, 2020. ArXiv preprint arXiv:2002.07741.
- 20 Leonard CG Rogers and Luitgard AM Veraart. Failure and rescue in an interbank network. *Management Science*, 59(4):882–898, 2013.
- 21 Steffen Schuldenzucker and Sven Seuken. Portfolio compression in financial networks: Incentives and systemic risk. In *Proceedings of the 21st ACM Conference on Economics and Computation*, EC ’20, page 79, New York, NY, USA, 2020. Association for Computing Machinery.
- 22 Steffen Schuldenzucker, Sven Seuken, and Stefano Battiston. Clearing payments in financial networks with credit default swaps. In *Proceedings of the 2016 ACM Conference on Economics and Computation*, EC ’16, pages 759–759, New York, NY, USA, 2016. ACM.

- 23 Steffen Schuldenzucker, Sven Seuken, and Stefano Battiston. Finding Clearing Payments in Financial Networks with Credit Default Swaps is PPAD-complete. In *8th Innovations in Theoretical Computer Science Conference (ITCS 2017)*, volume 67 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 32:1–32:20, Dagstuhl, Germany, 2017. Schloss Dagstuhl–Leibniz-Zentrum für Informatik.
- 24 Stefania Vitali, James B. Glattfelder, and Stefano Battiston. The network of global corporate control. *PloS one*, 6(10):1–6, 2011.

# No Quantum Speedup over Gradient Descent for Non-Smooth Convex Optimization

**Ankit Garg**

Microsoft Research India, Bangalore, India  
garga@microsoft.com

**Robin Kothari**

Microsoft Quantum and Microsoft Research, Redmond, WA, USA  
robin.kothari@microsoft.com

**Praneeth Netrapalli**

Microsoft Research India, Bangalore, India  
praneeth@microsoft.com

**Suhail Sherif**

Microsoft Research India, Bangalore, India  
Tata Institute of Fundamental Research, Mumbai, India  
suhail.sherif@gmail.com

---

## Abstract

We study the first-order convex optimization problem, where we have black-box access to a (not necessarily smooth) function  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  and its (sub)gradient. Our goal is to find an  $\epsilon$ -approximate minimum of  $f$  starting from a point that is distance at most  $R$  from the true minimum. If  $f$  is  $G$ -Lipschitz, then the classic gradient descent algorithm solves this problem with  $O((GR/\epsilon)^2)$  queries. Importantly, the number of queries is independent of the dimension  $n$  and gradient descent is optimal in this regard: No deterministic or randomized algorithm can achieve better complexity that is still independent of the dimension  $n$ .

In this paper we reprove the randomized lower bound of  $\Omega((GR/\epsilon)^2)$  using a simpler argument than previous lower bounds. We then show that although the function family used in the lower bound is hard for randomized algorithms, it can be solved using  $O(GR/\epsilon)$  quantum queries. We then show an improved lower bound against quantum algorithms using a different set of instances and establish our main result that in general even quantum algorithms need  $\Omega((GR/\epsilon)^2)$  queries to solve the problem. Hence there is no quantum speedup over gradient descent for black-box first-order convex optimization without further assumptions on the function family.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Quantum computation theory; Theory of computation  $\rightarrow$  Convex optimization

**Keywords and phrases** Quantum algorithms, Gradient descent, Convex optimization

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.53

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/2010.01801>.

**Acknowledgements** We thank Sébastien Bubeck, Ronald de Wolf, and András Gilyén for helpful conversations about this work. RK thanks Vamsi Pritham Pingali for many helpful conversations about multivariable calculus. We would also like to thank Matthias Kleinmann for asking us some questions about Lemma 17 that helped us improve its presentation.

## 1 Introduction

The classic gradient descent algorithm, first proposed by Cauchy in 1847, is a popular algorithm for minimizing functions in high-dimensional spaces. For some problems, such as the case of convex function minimization that we consider in this paper, gradient descent



© Ankit Garg, Robin Kothari, Praneeth Netrapalli, and Suhail Sherif;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 53; pp. 53:1–53:20



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



provably converges to the function’s global minimum. For other problems, such as finding good parameters of a deep neural network, gradient descent does not necessarily converge to a global minimum, and yet it has remarkable performance in practice.

Given the algorithm’s popularity, it is interesting to ask if gradient descent can be sped up on a quantum computer. However, it’s not obvious how to formalize this question since it’s not clear what it means for a quantum algorithm to speed up a given classical algorithm. For example, the best known classical algorithm for integer factorization is the general number field sieve (GNFS). Does Shor’s quantum algorithm for integer factorization speed up GNFS, or is it simply a different algorithm that solves the same problem?

One way to formalize the question *Can quantum computers speed up gradient descent?* is to consider a computational problem that is provably solved by gradient descent, and for which gradient descent is optimal among all classical algorithms. We can then ask if quantum algorithms can solve this problem faster than gradient descent. The second condition, that gradient descent is optimal among classical algorithms, is required since otherwise quantum computers would trivially be able to outperform gradient descent by using the best classical algorithm.

Fortunately, there is a canonical optimization task that is solved optimally by gradient descent: convex optimization with black-box first-order oracles. A more thorough introduction to the theory of black-box convex optimization can be found in the textbooks by Nemirovsky and Yudin [22], Nesterov [23, 24], and the monograph by Bubeck [11].

## 1.1 First-order convex optimization

Let’s start with the unconstrained convex minimization problem for a convex function  $f : \mathbb{R}^n \rightarrow \mathbb{R}$ . Here we want to find an  $x \in \mathbb{R}^n$  that’s  $\epsilon$ -close to minimizing the function  $f$ . More precisely, if we let  $x^* := \operatorname{argmin}_{x \in \mathbb{R}^n} f(x)$ , then our goal is to find any  $x \in \mathbb{R}^n$  such that  $f(x) - f(x^*) \leq \epsilon$ .

To obtain algorithms that are very general, this problem is often studied in the setting of black-box optimization. Here we do not assume any particular structure of the function  $f$  (e.g., that  $f$  is a low-degree polynomial), and only assume that we have some efficient method of computing  $f$  by an algorithm or circuit. In other words, we view  $f$  as a black box.

If we only had access to a black-box computing  $f$ , this would be zeroth-order optimization. In first-order optimization, we additionally assume we can also compute the gradient of  $f$ , or more precisely, since the gradient may not exist, we assume we can compute some subgradient of  $f$  (defined in Section 2). We call this oracle the *first-order oracle* and denote it by  $\mathcal{FO}(f)$ . In this work we consider arbitrary convex functions that are not necessarily smooth,<sup>1</sup> and so we cannot assume that the gradient exists. Our goal is to solve the function minimization problem while minimizing the number of calls or queries to the black boxes for  $f$  and some subgradient of  $f$ .

One might wonder why we consider queries to  $f$  and the subgradient of  $f$  to cost the same. This assumption is justified in many practical situations because of the *cheap gradient principle* [16], which says that “the cost to evaluate the gradient  $\nabla f$  is bounded above by a small constant times the cost to evaluate the function itself.” This provably holds in many models of computation; E.g., for arithmetic circuits over  $+$  and  $\times$ , it can be proved that the complexity of computing the gradient is at most 5 times the complexity

---

<sup>1</sup> In the optimization literature, a *smooth function* is a function that is differentiable everywhere in its domain, so the gradient is well defined, and whose gradient has bounded Lipschitz constant.

of computing  $f$  [6]. The conversion of source code computing  $f$  to code computing  $\nabla f$  can often be done automatically in many programming languages, and such methods are called *automatic differentiation* or *algorithmic differentiation* [16]. These same principles essentially carry over to the computation of subgradients [18]. In the quantum setting, there is additional motivation to assume that a function and its gradient cost roughly the same since we can obtain the gradient (or a subgradient) of a function from a black-box computing the function, as shown in a sequence of papers that make increasingly weaker assumptions on the function oracle [17, 15, 13, 3].

Now that we have black-box access to  $f$  and  $\mathcal{FO}(f)$ , we also need a starting point  $x_0 \in \mathbb{R}^n$  to begin our search for a minimum. We require this to be an input, and the complexity will depend on how close this is to  $x^*$ , since otherwise the interesting portion of the function where the minimum is achieved might be hiding in some small corner of  $\mathbb{R}^n$  that we cannot efficiently locate with only black-box access. Since we can easily shift the function by a fixed vector, without loss of generality we assume  $x_0 = \vec{0}$  is the origin. Let the distance between  $x_0 = \vec{0}$  and  $x^*$  be  $R := \|x^*\|$ .<sup>3</sup> For convenience, we will assume that  $R$  is part of the input as well, although this can be relaxed by binary searching for the correct value of  $R$ .

Finally, it is also reasonable that the complexity of our algorithms depend on how quickly  $f$  can change, since the value of  $f$  at some point only constrains its values at nearby points if the function does not change too rapidly. Let  $G$  be an upper bound on the Lipschitz constant of  $f$  (defined in Section 2), and we assume this is part of the input as well.

We are now ready to formally define the first-order convex minimization problem in the black-box setting. We use  $B(x, R) := \{y : \|x - y\| \leq R\}$  to denote an  $\ell_2$ -ball of radius  $R$  around  $x$ .

► **Problem 1** (First-order convex minimization). *Let  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  have Lipschitz constant at most  $G$  on  $B(\vec{0}, R)$ , and let*

$$x^* := \operatorname{argmin}_{x \in B(\vec{0}, R)} f(x). \quad (1)$$

*Then given  $n$ ,  $G$ ,  $R$ , and  $\epsilon > 0$ , the goal is to output a solution  $x \in B(\vec{0}, R)$  such that  $f(x) - f(x^*) \leq \epsilon$  while minimizing the number of queries to  $f$  and  $\mathcal{FO}(f)$ .<sup>4</sup>*

Note that we allow algorithms to query the function and gradient oracles at any point in  $\mathbb{R}^n$  even though the domain we are minimizing over is  $B(\vec{0}, R)$ . This only makes our lower bounds stronger, and the algorithms discussed in this paper never query the oracles outside the domain.

As we discuss in Section 2, although the problem seems to involve 4 parameters, the parameters  $G$ ,  $R$ , and  $\epsilon$  are not independent since we can rescale the input and output spaces of  $f$  and assume  $G = 1$  and  $R = 1$  without loss of generality. Thus any upper or lower bound on the complexity of this problem will be a function of  $n$  and  $GR/\epsilon$ .

## 1.2 Classical algorithms for first-order convex minimization

Gradient descent, or in this case subgradient descent, is a simple algorithm that starts from a point  $x_0$  and takes a small step (governed by a step size  $\eta$ ) in the opposite direction of the subgradient returned at  $x_0$ . Intuitively this brings us closer to the minimum since we are stepping in the direction where  $f$  decreases the most.

<sup>2</sup> If  $x^*$  is not unique, we can let  $R$  be the distance between  $x_0$  and the closest  $x^*$  to it.

<sup>3</sup> Throughout this paper  $\|\cdot\|$  always denotes the standard  $\ell_2$  norm in  $\mathbb{R}^n$  defined as  $\|z\| := \sqrt{\sum_i z_i^2}$ .

<sup>4</sup> For simplicity, we assume that these oracles output real numbers to arbitrarily many bits of precision. Since the main results of this paper are lower bounds, this only makes our results stronger.

We can now describe the performance of subgradient descent for Problem 1. Since this is a constrained optimization problem, we use the projected subgradient descent algorithm, which is subgradient descent with the added step of projecting the current vector back onto the ball  $B(\vec{0}, R)$  after every step.

► **Theorem 2** (Complexity of projected subgradient descent). *The projected subgradient descent algorithm solves Problem 1 using  $(GR/\epsilon)^2$  queries to  $f$  and  $\mathcal{FO}(f)$ .*

A proof of this is included in the full version of this paper. Observe that the query complexity of this algorithm, the number of queries made by the algorithm, is independent of  $n$ .<sup>5</sup> This is quite surprising at first and partly explains why gradient descent and its variants are popular in high-dimensional applications. More generally, we call such algorithms *dimension-independent algorithms*.

There also exist dimension-dependent algorithms for Problem 1 that work well when  $n$  is small. For example, the center of gravity method [11] solves this problem with  $O(n \log(GR/\epsilon))$  queries, which is very reasonable when  $n$  is small (and the algorithm is very efficient in terms of  $\epsilon$ ). In this work we focus on dimension-independent algorithms and assume that  $n$  is polynomially larger than the other parameters in the problem.

When  $n$  is large, we cannot improve over projected subgradient descent (Theorem 2) using any deterministic or randomized algorithm. We reprove the (well known) optimality of this algorithm among deterministic and randomized algorithms. This result is presented in Section 3.

► **Theorem 3** (Randomized lower bound). *For any  $G$ ,  $R$ , and  $\epsilon$ , there exists a family of convex functions  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  with  $n = O((GR/\epsilon)^2)$ , with Lipschitz constant at most  $G$  on  $B(\vec{0}, R)$ , such that any classical (deterministic or bounded-error randomized) algorithm that solves Problem 1 on this function family must make  $\Omega((GR/\epsilon)^2)$  queries to  $f$  or  $\mathcal{FO}(f)$  in the worst case.*

This lower bound on query complexity has been shown in several prior works [22, 26, 12], but we believe our proof is simpler and the dimension  $n$  required in our proof seems to be smaller than that in prior works. Note that while several expositions of gradient descent prove the lower bound for deterministic algorithms, very few sources establish a lower bound against randomized algorithms.

Our lower bound uses the following hard family of functions: For any  $z \in \{-1, +1\}^n$ , let  $f_z(x_1, \dots, x_n) = \max_{i \in [n]} z_i x_i$ ,<sup>6</sup> where  $n = O(1/\epsilon^2)$ . These functions are convex with Lipschitz constant 1. We show that finding an  $\epsilon$ -approximate minimum within  $B(\vec{0}, 1)$  requires  $\Omega(n)$  queries to the oracles. We establish the lower bound by showing that with high probability, every query of a randomized algorithm only reveals  $O(1)$  bits of information about the string  $z$ , but an  $\epsilon$ -approximate solution to this problem allows us to reconstruct the string  $z$ , which has  $n$  bits of information.

### 1.3 Quantum algorithms for first-order convex minimization

We then turn to quantum algorithms for solving Problem 1. At first, it might seem that since gradient descent is a sequential, adaptive algorithm where each step depends on the previous one, there is little hope of quantum algorithms outperforming gradient descent.

<sup>5</sup> Of course, the time complexity of implementing this algorithm will be at least linear in  $n$  since each query to either oracle requires us to manipulate a vector of length  $n$ .

<sup>6</sup> We use  $[n]$  to denote the set of positive integers less than or equal to  $n$ , i.e.,  $[n] := \{1, \dots, n\}$ .

On the other hand, consider the hard family of functions described above that witnesses the classical randomized lower bound in Theorem 3. While this is hard for classical algorithms, we show in Section 3.1 that there is a quantum algorithm that solves the problem on this family obtaining a quadratic speedup over any classical algorithm (and in particular, over gradient descent).

► **Theorem 4** (Quantum algorithm for classically hard function family). *There is a quantum algorithm that solves Problem 1 on the class of functions that appear in the classical lower bound of Theorem 3 using  $O(GR/\epsilon)$  queries to the oracle for  $f$ .*

Notably, unlike most quadratic speedups in quantum computing, the source of this quadratic speedup is not Grover’s algorithm or amplitude amplification. Theorem 4 uses Belovs’ quantum algorithm for learning symmetric juntas, which is constructed by exhibiting a feasible solution to the dual semidefinite program of the negative-weights adversary bound [7].

Now that we have shown a quadratic quantum speedup on a family of instances known to be hard for classical algorithms, there is some hope that quantum algorithms may provide some speedup for the general first-order convex minimization problem. Alas, our next result (established in Section 4), which is our main result, shows that this is not the case, and quantum algorithms cannot in general yield a speedup over classical algorithms for first-order convex minimization.

► **Theorem 5** (Quantum lower bound). *For any  $G$ ,  $R$ , and  $\epsilon$ , there exists a family of convex functions  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  with  $n = \widetilde{O}((GR/\epsilon)^4)$ , with Lipschitz constant at most  $G$  on  $B(\vec{0}, R)$ , such that any quantum algorithm that solves Problem 1 with high probability on this function family must make  $\Omega((GR/\epsilon)^2)$  queries to  $f$  or  $\mathcal{FO}(f)$  in the worst case.*

Our lower bound uses ideas from the lower bound against parallel randomized algorithms recently established by Bubeck, Jiang, Lee, Li, and Sidford [12].

At a high level, the hard family of functions used in the randomized lower bound does not work for quantum algorithms because although classical algorithms can only learn  $O(1)$  bits of information per query, quantum algorithms can make queries in superposition and learn a little information about many bits simultaneously. We remedy this by choosing a new family of functions in which with high probability, no matter what query the quantum algorithm makes, the oracle’s response is essentially the same. This allows us to control what the quantum algorithm learns per query, but now the instance is more complicated and the quantum algorithm learns  $O(n)$  bits of information per query. Since the final output of the algorithm is a vector in  $\mathbb{R}^n$ , we cannot use the argument used before that simply compared the information learned per query to the total information that needs to be learned. Instead we use the venerable hybrid argument [8] to control what the quantum algorithm learns and show that it cannot find an  $\epsilon$ -approximate solution to the minimization problem.

## 1.4 Related work

Classically, there is a long history of the study of oracle complexity (also known as black-box complexity or query complexity) for deterministic and randomized algorithms for non-smooth and smooth convex optimization. The setting considered in this paper, first-order convex optimization, where the algorithm has query access to the function value and the gradient, is very well studied. This topic is too vast to survey here, but we refer the reader to [22, 23, 24, 11] for more information about upper and lower bounds that can be shown in this setting.

There also has been work in the classical parallel setting, where in each round the algorithm is allowed to query polynomially many points and the goal is to minimize the number of rounds [21, 4, 14, 12]. Our work is most closely related to this setting and borrows many ideas from these works. Although quantum algorithms and parallel classical algorithms are incomparable in power, the constructions used to thwart parallel classical algorithms in these papers also help with showing quantum lower bounds.

In the quantum setting, there has been some work on convex optimization in the oracle model. There is also work on quantum gradient descent not in the oracle model. For example, one situation studied is where the dimension  $n$  of the optimization space is very large and the vectors are encoded in quantum states of dimension  $\log n$ . See [25, 19] and the references therein for more information. Another setting is the work on semidefinite programming, an important special case of convex optimization, but these algorithms exploit the specific structure of semidefinite programs [10, 2, 9, 1] and are not directly related to our work.

While in the classical setting, in general, a function value oracle is weaker than a gradient oracle, this is not the case in the quantum setting. Given a function value oracle, one can get a gradient oracle quite efficiently (with an  $\tilde{O}(1)$  overhead) [17, 15, 3, 13]. A similar result also holds for simulating a separation oracle given a membership oracle for convex bodies [3, 13]. As discussed before, our focus in this paper is to see if quantum algorithms can outperform classical algorithms when given a function oracle and gradient oracle since in many relevant settings, gradient computation is cheap in the classical case as well.

The most related works are the papers by Chakrabarti, Childs, Li, and Wu [13] and van Apeldoorn, Gilyén, Gribling, and de Wolf [3]. These papers establish very similar results so we cover them together. These papers study the problem of black-box convex optimization, and their results are phrased in the slightly different language of membership and separation oracles, but this is not the main difference between their work and our work. Indeed, it is possible to recast our problem in their setting (see the discussion in the introduction in [3] for how to do this). The main difference is that their algorithms are dimension-dependent and have complexities that depend on  $n$ , whereas we're working in the parameter regime where  $n$  is large and so we seek algorithms that are independent of  $n$ .

Specifically, [13] and [3] consider the problem of minimizing a linear function over a convex body given via a membership or separation oracle. A membership oracle for a convex body tells us whether a given point  $x$  is in the convex body and a separation oracle in addition when  $x$  is not in the body outputs a hyperplane that separates  $x$  from the convex body. Classically, the problem of outputting an  $\epsilon$ -approximate solution can be solved with  $O(n^2 \text{polylog}(\cdot))$  queries to a membership oracle [20], where we are suppressing polylogarithmic dependence on several parameters (including  $\epsilon$ ). These two papers show a quantum algorithm that makes only  $O(n \text{polylog}(\cdot))$  membership queries. The key technical component of this is a construction of a separation oracle from a membership oracle with only polylogarithmic overhead. To do this, they first show how to obtain an approximate subgradient oracle from a function oracle with only polylogarithmic overhead.

There are also several lower bounds shown in these papers. In [3], the authors prove that quantum algorithms do not give any advantage over classical algorithms in the setting where we are not given a point inside the convex body to start with. This setting is not directly comparable to our setting, as far as we are aware. In the setting where we do know a point inside the convex body, which is very similar to our setting, [3, 13] prove a lower bound of  $\Omega(\sqrt{n})$ , which is quadratically worse than their algorithm. While, in general, their results are incomparable to our results, one specific comparison to our results is that [13, Theorem 3.3] essentially shows a  $\tilde{\Omega}(\min\{GR/\epsilon, \sqrt{n}\})$  lower bound on the number of oracle

calls to a function value oracle for the setting in Problem 1.<sup>7</sup> Note that this is quadratically worse than our tight lower bound (Theorem 5) in the dimension-independent setting (i.e., when the dimension  $n$  is large compared to  $GR/\epsilon$ ).

## 1.5 Paper organization and summary of contributions

We first present some preliminaries on convex optimization in Section 2. In Section 3 we reprove the lower bound for randomized algorithms (Theorem 3) using a simpler argument compared to prior works. In Section 3.1, we show that quantum algorithms can solve the hard instance from Theorem 3 faster than randomized algorithms, obtaining a quadratic speedup (Theorem 4). In Section 4, we present a different hard instance and show our main result that quantum algorithms cannot obtain any speedup over gradient descent for the first-order convex optimization problem (Theorem 5). We conclude with open problems in Section 5.

## 2 Convex optimization preliminaries

As described in the introduction, we are interested in approximately minimizing a convex function  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  on some closed convex set  $\mathcal{K} \subseteq \mathbb{R}^n$ . A function  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  is convex if for all  $x, y \in \mathbb{R}^n$  and  $t \in [0, 1]$ ,

$$tf(x) + (1-t)f(y) \geq f(tx + (1-t)y). \quad (2)$$

A set  $\mathcal{K} \subseteq \mathbb{R}^n$  is convex if the line segment joining two points in  $\mathcal{K}$  is also contained in  $\mathcal{K}$ . We will consider convex sets of bounded size, and specifically let  $2R$  be the diameter of  $\mathcal{K}$ , i.e.,

$$\max_{x, y \in \mathcal{K}} \|x - y\| \leq 2R, \quad (3)$$

where  $\|z\| := \sqrt{\sum_i z_i^2}$  is the Euclidean norm.

It turns out that the query complexity of first-order convex optimization depends only on  $R$  no matter how complicated the set  $\mathcal{K}$  happens to be. However, to obtain an algorithm with efficient time complexity we require that the set  $\mathcal{K}$  be simple enough that we can efficiently implement a projection operator for  $\mathcal{K}$ . This means given any  $y \in \mathbb{R}^n$ , we can efficiently compute  $\mathcal{P}_{\mathcal{K}}(y) \in \mathcal{K}$ , which satisfies  $\|\mathcal{P}_{\mathcal{K}}(y) - y\| = \min_{z \in \mathcal{K}} \|z - y\|$ . Since the main result of this paper is a lower bound, our lower bound is stronger if shown for a simple convex set  $\mathcal{K}$ . So throughout this paper we work with the set  $\mathcal{K} = B(\vec{0}, R)$ , the  $\ell_2$ -ball of radius  $R$  around the origin.

In the model of first-order black-box optimization, we have access to a black-box that computes the function  $f$  on any input  $x \in \mathbb{R}^n$ . In addition to this, we also have a *first-order oracle*,  $\mathcal{FO}(f)$ , which when queried at any point  $x \in \mathbb{R}^n$  returns some vector  $g_x \in \mathbb{R}^n$  that satisfies for all  $y \in \mathbb{R}^n$ ,

$$f(y) \geq f(x) + \langle g_x, y - x \rangle. \quad (4)$$

Since  $f$  is convex, it is known that such a vector  $g_x$  exists for all  $x \in \mathbb{R}^n$  [23]. Any vector  $g_x$  satisfying (4) is called a subgradient of  $f$  at  $x$ , and the set of all subgradients at  $x$  is called the subdifferential at  $x$  and denoted by  $\partial f(x)$ . If  $f$  is differentiable at  $x$  then  $g_x$  is unique and equal to  $\nabla f(x)$ , the gradient of  $f$  at  $x$ , defined as

<sup>7</sup> This is equivalent to our setting, where we have a function value and gradient oracle, due to their results.

$$\nabla f(x) := \left( \frac{\partial f(x)}{\partial x_1}, \dots, \frac{\partial f(x)}{\partial x_n} \right). \quad (5)$$

We will say that the function  $f$  has Lipschitz constant at most  $G$  in  $\mathcal{K}$  if  $\|g_x\| \leq G$  for every  $x \in \mathcal{K}$ .<sup>8</sup>

As described in Problem 1, we are interested in algorithms that take as inputs the parameters  $n$ ,  $G$ ,  $R$ , and  $\epsilon > 0$ , and have access to  $f$  and a first-order oracle  $\mathcal{FO}(f)$ , and output  $x \in B(\vec{0}, R)$  such that  $f(x) - f(x^*) \leq \epsilon$ , where  $x^* := \operatorname{argmin}_{x \in B(\vec{0}, R)} f(x)$ .

In the quantum setting, we have quantum analogues of these oracles. There is a straightforward generalization of any oracle to the quantum setting, which makes the classical oracle reversible and then allows queries in superposition to this oracle. This quantum generalization of the oracle is justified by the fact that if we had a classical circuit or algorithm computing a function  $f$ , then it is possible in a completely black-box manner to construct the quantum oracle corresponding to  $f$ , and this oracle would then support superposition queries. We discuss quantum oracles in more detail in Section 4, but for now it is sufficient to consider them as computing the same functions as the classical oracles, except that they can additionally be queried in superposition.

Note that it is sufficient to consider the special case of the problem where  $G = R = 1$ . While this seems like a special case, given an  $f$  and  $\mathcal{K}$  with Lipschitz constant  $G$ , radius  $R$ , and optimization accuracy  $\epsilon$ , we can instead minimize  $\hat{f}(x) := \frac{1}{GR} f(Rx)$  over  $\hat{\mathcal{K}} := \mathcal{K}/R$ , which have Lipschitz constant and radius 1 up to an accuracy of  $\frac{\epsilon}{GR}$ . So we consider  $G = R = 1$  without loss of generality, or for general  $G$  and  $R$ , the complexity must be a function of  $GR/\epsilon$ .

The query complexity of an algorithm that solves Problem 1 is the maximum number of oracle calls it makes for fixed values of  $n$ ,  $G$ ,  $R$ , and  $\epsilon$ , where the maximum is taken over all convex functions  $f$  with Lipschitz constant at most  $G$ , and all first order oracles  $\mathcal{FO}(f)$  for  $f$  (i.e., the algorithm must work for any choice of first-order oracle that correctly outputs some subgradient of  $f$  at  $x$ ). As discussed, the query complexity must be a function of  $n$  and  $GR/\epsilon$ . Furthermore, since we're interested in dimension-independent algorithms, we study algorithms that only depend on  $GR/\epsilon$  and not on  $n$ .

Given a class of algorithms, such as deterministic, randomized, or quantum algorithms, the query complexity of first-order Lipschitz convex optimization for that class of algorithms is the minimum query complexity of any algorithm in that class that solves Problem 1.

As we will see in the next section, the randomized query complexity of this problem (and hence the deterministic query complexity) is at least  $\Omega((GR/\epsilon)^2)$  in the dimension-independent setting.

### 3 Randomized Lower Bound

In this section, we prove a lower bound for randomized first-order methods for non-smooth convex optimization, restated here for convenience:

<sup>8</sup> This is slightly different from the usual definition of the Lipschitz constant where we would say  $f$  is  $G$ -Lipschitz in  $\mathcal{K}$  if for all  $x, y \in \mathcal{K}$ ,  $|f(x) - f(y)| \leq G\|x - y\|$ . Our definition is the same as requiring the function  $f$  to be  $G$ -Lipschitz according to this definition in an open set that contains  $\mathcal{K}$ .



► **Theorem 3** (Randomized lower bound). *For any  $G$ ,  $R$ , and  $\epsilon$ , there exists a family of convex functions  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  with  $n = O((GR/\epsilon)^2)$ , with Lipschitz constant at most  $G$  on  $B(\vec{0}, R)$ , such that any classical (deterministic or bounded-error randomized) algorithm that solves Problem 1 on this function family must make  $\Omega((GR/\epsilon)^2)$  queries to  $f$  or  $\mathcal{FO}(f)$  in the worst case.*

This randomized lower bound is known and multiple proofs can be found in the literature [22, 26]. Our proof is elementary and we did not find it written anywhere, although it is conceptually similar to the one in [22], and so we include it here for completeness. Our proof also has the dimension  $n = \Theta(1/\epsilon^2)$ , without any log factors, which is the best possible. As far as we are aware, the previous proofs (for randomized algorithms) required larger dimension than this.<sup>9</sup> As we will see later, the family of instances used is also interesting because we can get a quantum speedup for it, because of which we have to look at other instances to prove the quantum lower bound.

We can now define the family of convex functions used in the lower bound. For any  $\epsilon > 0$ , we set  $n = \lfloor .9/\epsilon^2 \rfloor$  and look at the following class of functions.

► **Definition 6.** *Let  $z \in \{-1, +1\}^n$ . Let  $f_z : \mathbb{R}^n \rightarrow \mathbb{R}$  be defined as*

$$f_z(x_1, \dots, x_n) = \max_{i \in [n]} z_i x_i. \quad (6)$$

Each such function is convex since it is a maximum of convex functions [23, Theorem 3.1.5]. Note that if  $f_z(x) = z_i x_i$  for some  $i \in [n]$ , then  $z_i e_i$  is a subgradient of  $f_z$  at  $x$  (since  $f_z(x) + \langle z_i e_i, y - x \rangle = z_i y_i \leq f_z(y)$ ). Hence the function is 1-Lipschitz. We can also see that within the unit ball the function is minimized at the point

$$x^* = \frac{-1}{\sqrt{n}} \sum_{i \in [n]} z_i e_i, \quad (7)$$

and  $f_z(x^*) = -1/\sqrt{n}$ . Clearly given  $x^*$  we can recover  $z$  from it. We now show  $z$  can even be recovered from an  $\epsilon$ -approximate minimum of  $f_z$ .

► **Lemma 7.** *Let  $x$  be such that  $f_z(x) - f_z(x^*) \leq \epsilon$ . Then we can recover  $z \in \{-1, +1\}^n$  from  $x \in \mathbb{R}^n$ .*

**Proof.** Let  $s_x \in \{-1, +1\}^n$  be the vector with  $(s_x)_i = \text{sign}(x_i)$ , where  $\text{sign}(a) = +1$  if  $a \geq 0$  and  $\text{sign}(a) = -1$  otherwise. We claim that  $z = -s_x$ . Toward a contradiction, if  $(s_x)_i \neq -z_i$  for some  $i$ , then  $(s_x)_i = z_i$ , since these only take values in  $\{-1, +1\}$ . In this case,  $x_i$  and  $z_i$  agree in sign, and hence  $f_z(x) \geq z_i x_i \geq 0$ . Since  $\epsilon < 1/\sqrt{n}$  (because of our choice of  $n$  above) the point  $x$  cannot satisfy  $f_z(x) - f_z(x^*) \leq \epsilon$ . ◀

Since this function is not differentiable everywhere, for our lower bound we need to specify the behavior of the subgradient oracle on all inputs. The function is not differentiable only at  $x \in \mathbb{R}^n$  where the maximum is achieved at multiple indices. In this case, the subgradient oracle responds as if the maximum was achieved on the smallest such index  $i$ , i.e., it responds with  $z_i e_i$ . Note that for this function, querying the subgradient oracle allows us to simulate a call to the function oracle as well, since the response is  $z_i e_i$  for the index  $i$  that achieves the maximum, so the function evaluates to  $z_i x_i$  at that point, which we can compute since we know  $x$ . So we can assume without loss of generality that an algorithm only queries the subgradient oracle.

<sup>9</sup> The result of [26] requires dimension roughly  $1/\epsilon^8$ . We also believe the result in [22] requires a larger dimension than our claim.

Now that the problem is fully specified, we will show that any randomized optimization algorithm using the function oracle and this subgradient oracle will require  $\Omega(n)$  queries in order to solve Problem 1 with a constant probability of success.

The following will be the crux of the lower bound. Let  $I \subseteq [n]$ . We say a distribution  $\mathcal{D}$  over  $\{-1, +1\}^n$  is  $I$ -fixed if for  $z \sim \mathcal{D}$  the random variable  $z_I$  is fixed and  $z_{\bar{I}}$  is uniform over  $\{-1, +1\}^{\bar{I}}$ .

► **Lemma 8.** *Let  $z$  be distributed according to an  $I$ -fixed distribution. Let  $x$  be an arbitrary query made to the  $f_z$  oracle. After one query to the subgradient oracles, the conditional distribution on  $z$  given the answer is  $I'$ -fixed with  $I \subseteq I'$  and  $\mathbb{E}[|I'|] \leq |I| + 2$ .*

**Proof.** Let  $x$  be the algorithm's query. The index  $i$  that achieves the maximum in the definition of  $f_z(x)$  can be computed as follows. Let  $i_1, \dots, i_n$  be the ordering of the indices 1 to  $n$  in decreasing order of  $|x_i|$ , with ties broken with the natural ordering on integers. The oracle outputs  $f_z(x) = z_{i_j} x_{i_j}$  and chooses the subgradient  $z_{i_j} e_{i_j}$  where  $j$  is the smallest index for which  $x_{i_j}$  agrees in sign with  $z_{i_j}$ , and if no such index exists, then  $j = n$ .

Since  $f_z(x)$  can be computed given the subgradient  $z_{i_j} e_{i_j}$ , the only information obtained from a query is the prefix  $\{z_{i_k}\}_{k \leq j}$ . In other words, if the subgradient oracle responds with  $z_{i_j} e_{i_j}$ , then we have learned that for all indices  $k \leq j$ , we must have  $\text{sign}(x_i) = -z_i$ , but we have not learned any more since the oracle's output does not depend on the bits of  $z$  with index  $i_k$  with  $k > j$ . After this query, we know the bits  $z_{i_k}$  with  $k \leq j$ , but conditioned on these, the distribution on the remaining bits of  $z$  continues to be uniform. This is an  $I'$ -fixed distribution with  $I' = I \cup \{i_k\}_{k \leq j}$ . Intuitively,  $I'$  cannot be much larger than  $I$  since an index  $i_k$  is part of this set only if the algorithm correctly guessed the sign of  $z_{i_k}$  for this index and all indices with a smaller value of  $k$ . Since the initial distribution  $z$  was uniformly at random outside of  $I$  and  $x$  is fixed, the probability of correctly guessing the first index (according to the  $i_j$  ordering) that was not fixed is  $1/2$ , the probability of guessing the first two is  $1/4$  and so on. Thus the expected number of new entries fixed by one query is  $\sum_{k=1}^{n \setminus |I|} k \cdot \frac{1}{2^k} \leq 2$ . ◀

We can use this to show establish the final claim.

► **Lemma 9.** *Let  $z$  be sampled uniformly at random from  $\{-1, +1\}^n$ . If a randomized algorithm  $\mathcal{A}$  outputs an  $x$  with  $f_z(x) - f_z(x^*) \leq \epsilon$  with probability at least  $2/3$ , then its query complexity is at least  $n/3 - 1$ .*

**Proof.** When  $\mathcal{A}$  outputs a point  $x$ , we will require it to also query the oracle at  $x$  to see if it is indeed  $\epsilon$ -optimal. This can increase its query complexity by at most one. Let the query complexity of this modified  $\mathcal{A}$  be  $t$ . Whenever  $\mathcal{A}$  does output an  $\epsilon$ -optimal point, Lemma 7 implies that the conditional distribution on  $z$  is  $[n]$ -fixed. For each  $i \in [0, \dots, t]$ , let  $I_i$  be the random variable such that the distribution on  $z$  after  $i$  queries of  $\mathcal{A}$  is  $I_i$ -fixed (Lemma 8 implies that after any sequence of queries it will be an  $I$ -fixed distribution for some  $I$ ). Since  $z$  is sampled uniformly at random from  $\{-1, +1\}^n$ ,  $I_0 = \emptyset$ . And since we want the algorithm to succeed with probability at least  $2/3$ ,  $\mathbb{E}[|I_t|] \geq 2n/3$ .

However,  $|I_t| = \sum_{i=1}^t |I_i| - |I_{i-1}|$ , and it is a simple consequence of Lemma 8 that  $\mathbb{E}[|I_i| - |I_{i-1}|] \leq 2$  for all  $i$ . So by the linearity of expectation,  $\mathbb{E}[|I_t|] \leq 2t$  and hence  $t \geq n/3$ . ◀

This proves a lower bound of  $\Omega(1/\epsilon^2)$  on the randomized query complexity of first-order convex minimization for a function with  $G = R = 1$ . As noted earlier, this is without loss of generality and implies the more general bound in Theorem 3.

### 3.1 Quantum speedup

In this section we discuss Theorem 4, restated for convenience:

► **Theorem 4** (Quantum algorithm for classically hard function family). *There is a quantum algorithm that solves Problem 1 on the class of functions that appear in the classical lower bound of Theorem 3 using  $O(GR/\epsilon)$  queries to the oracle for  $f$ .*

The quantum speedup for the above class of functions relies on Belovs' quantum algorithm for Combinatorial Group Testing [7]. Belovs showed that given access to an oracle making OR queries to an  $n$ -bit string, the  $n$ -bit string can be learned in  $O(\sqrt{n})$  quantum queries. The proof of Theorem 4 then follows by noting that for any  $S \subseteq [n]$ , one can find out if there is an  $i \in S$  such that  $z_i = 1$  by querying  $x = \frac{1}{\sqrt{n}} \sum_{i \in S} e_i$ . Hence within  $O(\sqrt{n})$  queries, the entire string  $z \in \{-1, +1\}^n$  can be learned. This proof is provided in more detail in the full version of the paper.

A similar problem is also studied in [13], which they solve using a quantum algorithm for the search with wildcards problem, which is a special case of the combinatorial group testing problem studied by Belovs.

Note that this quantum algorithm is also essentially optimal for this problem and it is not hard to show an  $\Omega(\sqrt{n}/\log n)$  lower bound for quantum algorithms. A similar lower bound is shown in [13, Theorem 3.3], and a simpler sketch of such a lower bound is also given in the full version.

## 4 Quantum lower bound

In this section, we show that for any  $\epsilon$ , there exists a 1-Lipschitz family of functions such that any quantum algorithm that solves Problem 1 on the unit ball must make  $\frac{1}{100\epsilon^2}$  queries. In other words, there is no quantum first-order convex optimization algorithm that always outperforms the classical gradient descent algorithm described in Theorem 2. The function we will use was introduced by Nemirovsky and Yudin [22]. To show the quantum lower bound, we adapt to the quantum setting the lower bound strategy of Bubeck et al. [12] in the model of parallel algorithms.

We restate the main result proved in this section for convenience:

► **Theorem 5** (Quantum lower bound). *For any  $G$ ,  $R$ , and  $\epsilon$ , there exists a family of convex functions  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  with  $n = \tilde{O}((GR/\epsilon)^4)$ , with Lipschitz constant at most  $G$  on  $B(\vec{0}, R)$ , such that any quantum algorithm that solves Problem 1 with high probability on this function family must make  $\Omega((GR/\epsilon)^2)$  queries to  $f$  or  $\mathcal{FO}(f)$  in the worst case.*

We start by first proving a qualitatively similar, but simpler result with a larger value of  $n = \tilde{O}((GR/\epsilon)^6)$  in Section 4.4. If we only care about the optimality of gradient descent in the dimension-independent setting, this lower bound is sufficient. But if we also want to understand the trade-off between dimension-independent and dimension-dependent algorithms, then we would like to show this lower bound with as small a value of  $n$  as we can. In the full version of the paper, we improve the lower bound to achieve the value of  $n$  stated in this theorem.

### 4.1 Function family and basic properties

We start by defining the family of functions  $\mathcal{F} = \{f : \mathbb{R}^n \rightarrow \mathbb{R}\}$  that we use. The function family  $\mathcal{F}$  depends on the dimension  $n$  and two other parameters  $k$  and  $\gamma$ . Since the function family we choose depends on  $\epsilon$ , the parameters  $n$ ,  $k$ , and  $\gamma$  will be functions of  $\epsilon$ . Our choice of  $n$ ,  $k$ , and  $\gamma$  will become clear later, but for now we simply choose them as follows. Let

$$k := \frac{1}{100\epsilon^2} \implies \epsilon = \frac{1}{10\sqrt{k}} \quad \text{and} \quad \gamma := \frac{1}{10k^{3/2}} = 100\epsilon^3. \quad (8)$$

We choose  $n$  such that it satisfies

$$\gamma \geq 8\sqrt{\frac{\log n}{n}} \implies n := O\left(\frac{\log(1/\epsilon)}{\epsilon^6}\right) = \tilde{O}\left(\frac{1}{\epsilon^6}\right). \quad (9)$$

The discussion before Lemma 11 explains the choice of  $k$  and the discussion after Lemma 12 explains the choice of  $\gamma$ . For the dimension  $n$ , see the discussion at the beginning of Section 4.2.

We now define the function family for these specific choices of  $n$ ,  $k$ , and  $\gamma$ .

► **Definition 10** (Hard function family). *Let  $\mathcal{V} = \{(v_1, \dots, v_k) \mid \forall i, j, \in [k], \langle v_i, v_j \rangle = \delta_{ij}\}$  be the set of all  $k$ -tuples of orthonormal vectors in  $\mathbb{R}^n$ . Let the family of functions  $\mathcal{F} = \{f_V\}_{V \in \mathcal{V}}$  be defined as*

$$f_{(v_1, v_2, \dots, v_k)}(x) := \max_{i \in [k]} \{g_V^{(i)}(x)\}, \quad \text{where } g_V^{(i)}(x) := \langle v_i, x \rangle + (k-i)\gamma\|x\|. \quad (10)$$

We will show that any quantum algorithm that solves Problem 1 on the functions in this family must make  $k$  queries. As we will prove, informally what happens is each query of the quantum algorithm to the gradient oracle only reveals a single direction  $v_i$  to the algorithm. In fact, with very high probability the vectors are revealed in order, so that the algorithm first learns  $v_1$ , then  $v_2$ , and so on. As we will show in Lemma 12, any  $\epsilon$ -optimal solution must overlap significantly with all  $v_i$ , and thus any quantum algorithm must make  $k$  queries. Since we want to show an  $\Omega(1/\epsilon^2)$  bound, we choose  $k$  to be a small multiple of  $1/\epsilon^2$ , which explains our choice for  $k$  in Equation (8).

We now establish some basic properties of these functions.

► **Lemma 11** (Properties of  $f_V$ ). *For any  $V \in \mathcal{V}$ , let  $f_V$  and  $g_V^{(i)}$  be as in Definition 10. Then  $f_V$  is convex with Lipschitz constant at most  $1 + k\gamma \leq 2$  on  $B(\vec{0}, 1)$ , and*

$$\text{for } x \neq \vec{0}, \quad \nabla g_V^{(i)}(x) = v_i + (k-i)\gamma x/\|x\|, \quad \text{and} \quad (11)$$

$$\text{for } x = \vec{0}, \quad \partial g_V^{(i)}(\vec{0}) = \{v_i + (k-i)\gamma u \mid u \in B(\vec{0}, 1)\}, \quad \text{and} \quad (12)$$

$$\text{for any } x, \quad \partial f_V(x) = \text{ConvexHull}(\{u \in \partial g_V^{(i)}(x) \mid g_V^{(i)}(x) = f_V(x)\}), \quad (13)$$

where the convex hull of a set of vectors is the set of all convex combinations of vectors in the set. Lastly, for any  $\alpha > 0$ ,  $f_V(\alpha x) = \alpha f_V(x)$  and  $\partial f_V(\alpha x) = \partial f_V(x)$ .

**Proof.** For all  $V \in \mathcal{V}$ ,  $f_V : \mathbb{R}^n \rightarrow \mathbb{R}$  is convex. This follows because linear functions and norms are convex functions [23, Example 3.1.1], and the sum or maximum of convex functions is convex [23, Theorem 3.1.5].

Let us now compute the subgradients of  $g_V^{(i)}(x) = \langle v_i, x \rangle + (k-i)\gamma\|x\|$ . The linear function  $\langle v_i, x \rangle$  is differentiable and its gradient is simply  $v_i$ . The Euclidian norm  $\|x\|$  is differentiable everywhere except at  $x = \vec{0}$ . At  $x \neq \vec{0}$ , the gradient of  $\|x\|$  is  $x/\|x\|$  and at  $x = 0$ , the set of subgradients is  $B(\vec{0}, 1)$  [23, Example 3.1.5]. We also know that  $\partial(\alpha_1 f_1(x) + \alpha_2 f_2(x)) = \alpha_1 \partial f_1(x) + \alpha_2 \partial f_2(x)$  [23, Lemma 3.1.9], which gives us the expressions for the subgradients of  $g_V^{(i)}$ .

For a function that is the maximum of functions  $g_V^{(i)}$ , we know that the set of subgradients is simply the convex hull of subgradients of those  $g_V^{(i)}$  which achieve the maximum at the given point  $x$  [23, Lemma 3.1.10].

The Lipschitz constant of a function is the maximum norm of any subgradient of the function. Since any vector in  $\partial g_V^{(i)}$  has norm  $1 + k\gamma$ , and any vector in  $\partial f_V$  is the convex combination of vectors with norm at most  $1 + k\gamma$ , the Lipschitz constant of  $f_V$  is at most  $1 + k\gamma \leq 2$ .

Finally, it is easy to see from the definition of  $f_V$  that for  $\alpha > 0$ ,  $f_V(\alpha x) = \alpha f(x)$  since each term in the max gets multiplied by  $\alpha$ . For  $\partial f_V(\alpha x)$ , note that this is a convex combination of  $\partial g_V^{(i)}(\alpha x)$ , and these do not depend on  $\alpha$ . ◀

For convenience we work with this family of functions with Lipschitz constant at most 2 instead of 1, which doesn't change the asymptotic bounds since we could just divide every function  $f_V$  by 2.

The last property essentially says that querying the function or its subgradient on a scalar multiple of a vector  $x$  gives us only as much information as querying it on  $x$ . Thus we can assume that an algorithm only queries the oracles within the unit ball without loss of generality.

Now let us discuss the vector  $x^* \in B(\vec{0}, 1)$  that minimizes  $f_V(x)$  and vectors that  $\epsilon$ -approximately solve the minimization problem. First note that if  $\gamma$  were equal to 0, then the function would simply be  $\max_{i \in [k]} \langle v_i, x \rangle$ , which requires us to minimize the component of  $x$  in  $k$  different directions subject to it being a unit vector. The solution to this is simply  $\frac{-1}{\sqrt{k}} \sum_i v_i$ . Now  $-1/\sqrt{k} = -10\epsilon$ , so the overlap of  $x$  with each direction  $v_i$  is a large multiple of  $\epsilon$ . So even an  $\epsilon$ -approximate solution must have reasonable overlap with each of the vectors  $v_i$ . Specifically, each overlap must be at least  $-9\epsilon$ . Now in our function  $f_V$  the term  $\gamma$  is not 0, but that term at most perturbs the function by  $k\gamma = \epsilon$ , which again is much smaller than  $10\epsilon$ , and thus even approximate solutions must have significant overlaps with all  $v_i$ . We formalize these properties below.

► **Lemma 12** (Properties of the minimum). *For any  $V \in \mathcal{V}$ , let  $f_V : \mathbb{R}^n \rightarrow \mathbb{R}$  be the function in Definition 10 and let  $x^* := \operatorname{argmin}_{x \in B(\vec{0}, 1)} f_V(x)$ . Then  $f_V(x^*) \leq -9\epsilon$ . Furthermore, any  $x \in \mathbb{R}^n$  that satisfies  $|f_V(x) - f_V(x^*)| \leq \epsilon$  must satisfy for all  $i \in [k]$ ,  $\langle v_i, x \rangle \leq -8\epsilon$ .*

**Proof.** Consider the vector  $y = \frac{-1}{\sqrt{k}} \sum_{i \in [k]} v_i$ . This is a vector in  $B(\vec{0}, 1)$ , satisfying  $f_V(y) \leq \frac{-1}{\sqrt{k}} + (k-1)\gamma \leq \frac{-1}{\sqrt{k}} + k\gamma = -10\epsilon + \epsilon = -9\epsilon$ , because we have  $10\epsilon = \frac{1}{\sqrt{k}}$  and  $k\gamma = \frac{1}{10\sqrt{k}} = \epsilon$ . Thus  $f_V(x^*) \leq f_V(y) \leq -9\epsilon$ .

Now consider any vector  $x$  with  $|f_V(x) - f_V(x^*)| \leq \epsilon$ , which implies  $f_V(x) \leq -8\epsilon$ . If  $\langle v_i, x \rangle > -8\epsilon$  for any  $i \in [k]$ , then  $f_V(x) \geq \langle v_i, x \rangle + (k-i)\gamma\|x\| > -8\epsilon$ , which is a contradiction. ◀

This result crucially uses the relation between  $\gamma$  and  $k$  and because we want  $k\gamma$  to be a constant factor (say 10) smaller than  $1/\sqrt{k}$ , this informs our choice of  $\gamma$  in Equation (8). Our choice of  $n$  in Equation (9) will be discussed in the next section.

## 4.2 Probabilistic facts about the function family

So far all the properties we have discussed of our function family hold for any  $V \in \mathcal{V}$ , but now we want to talk about a hard distribution over such functions. Specifically we want to talk about choosing a uniformly random (according to the Haar measure)  $V$  from the infinite set  $\mathcal{V}$ . It is easy to see how to sample a random  $V$  once we can sample unit vectors from a subspace. We start by choosing  $v_1$  to be a Haar random unit vector from  $\mathbb{R}^n$ , let  $v_2$  be a Haar random unit vector from  $\operatorname{span}(v_1)^\perp$ , and so on, until  $v_k$  is a Haar random unit vector in  $\operatorname{span}(v_1, v_2, \dots, v_{k-1})^\perp$ .

We can now discuss what determines our choice of  $n$ . By construction, the family of functions  $\mathcal{F}$  has the property that if the input vector  $x$  has equal inner product with all vectors  $v_i$ , then the maximum will be achieved uniquely on the first term  $i = 1$  because the additive term  $(k - i)\gamma\|x\|$  is largest for  $i = 1$ . Now what we want to ensure is that this property holds even when  $x$  does not have equal inner product with all  $v_i$ , but  $x$  is chosen uniformly at random from  $B(\vec{0}, 1)$ . Or equivalently, we want this property to hold when  $x$  is fixed, but the set  $V$  is chosen uniformly at random.

In either case, the inner product of  $x$  with a random unit vector  $v$  will be a random variable with mean 0 due to symmetry. But the expected value of  $|\langle v, x \rangle|^2$  for a random unit vector  $v$  is  $1/n$ , and in fact it will be tightly concentrated around  $1/n$ . The following proposition follows from [5, Lemma 2.2].

► **Proposition 13.** *Let  $x \in B(\vec{0}, 1)$ . Then for a random unit vector  $v$ , and all  $c > 0$ ,*

$$\Pr_v(|\langle x, v \rangle| \geq c) \leq 2e^{-nc^2/2}. \quad (14)$$

We choose  $\gamma$  so that it is very unlikely (polynomially small in  $n$ ) that the maximum is not achieved at  $i = 1$ . From Proposition 13, we see that the probability of any  $|\langle v_i, x \rangle|^2$  being larger than a constant multiple of  $\log n/n$  is inverse polynomially small. So it is sufficient to take  $\gamma^2$  to be a large constant multiple of  $\log n/n$  as in Equation (9).

In our lower bound we will need a slightly stronger result. We can show that if the vectors  $v_1, \dots, v_{t-1}$  are fixed (and hence known to the algorithm), and the remaining vectors  $v_t, \dots, v_k$  are chosen uniformly at random such that the set of vectors  $\{v_1, \dots, v_k\}$  is orthonormal, then the maximum will be achieved in the set  $[t]$  with high probability. This generalizes the previous claim, which is the case of  $t = 1$ , where none of the vectors were fixed.

► **Lemma 14 (Most probable argmax).** *Let  $1 \leq t \leq k$  be integers and  $\{v_1, \dots, v_{t-1}\}$  be a set of orthonormal vectors. Let  $\{v_t, \dots, v_k\}$  be chosen uniformly at random so that the set  $\{v_1, \dots, v_k\}$  is orthonormal. Then*

$$\forall x \in B(\vec{0}, 1) : \Pr_{v_t, \dots, v_k} \left( \max_{i \in [k]} \langle v_i, x \rangle + (k - i)\gamma\|x\| \neq \max_{i \in [t]} \langle v_i, x \rangle + (k - i)\gamma\|x\| \right) \leq \frac{1}{n^7}. \quad (15)$$

**Proof.** Let  $E_x$  denote the event whose probability we want to upper bound. Since  $E_x$  and  $E_{\alpha x}$ , for any  $\alpha \in [0, 1]$ , are the same event, we can assume without loss of generality that  $\|x\| = 1$ . If event  $E_x$  occurs, then it must hold that

$$\max_{i \in \{t+1, \dots, k\}} \langle v_i, x \rangle + (k - i)\gamma > \max_{i \in [t]} \langle v_i, x \rangle + (k - i)\gamma \geq \langle v_t, x \rangle + (k - t)\gamma. \quad (16)$$

We want to show that this event is very unlikely. To do so, let  $F_x$  be the event that for all  $i \in \{t, \dots, k\}$ ,  $\langle v_i, x \rangle \in [-\frac{\gamma}{2}, +\frac{\gamma}{2}]$ . Note that if  $F_x$  occurs, then the terms in the max are in decreasing order, and we have

$$\langle v_t, x \rangle + (k - t)\gamma \geq \langle v_{t+1}, x \rangle + (k - t - 1)\gamma \geq \dots \geq \langle v_{k-1}, x \rangle + \gamma \geq \langle v_k, x \rangle, \quad (17)$$

which contradicts Equation (16). Thus if  $E_x$  holds then the complement of  $F_x$ ,  $\bar{F}_x$  must hold, which means  $\Pr(E_x) \leq \Pr(\bar{F}_x)$ . So let us show that  $F_x$  is very likely.

The event  $\bar{F}_x$  holds only if there exists an  $i \in \{t, \dots, k\}$  such that  $\langle v_i, x \rangle \notin [-\frac{\gamma}{2}, +\frac{\gamma}{2}]$ . We can upper bound this probability for any particular  $i \in \{t, \dots, k\}$  using Proposition 13 and the fact that  $v_i$  is chosen uniformly at random from an  $n - t + 1$ -dimension ball. This probability is at most  $2e^{-(n-t+1)\gamma^2/8} = 2e^{-(n-t+1) \cdot 8 \frac{\log n}{n}} \leq 2 \cdot 2^{-8 \log n} = 2/n^8$ , with the inequality holding because  $n > 4t$ . The probability that this happens for any  $i$  is at most  $(k - t + 1) \leq k$  times this probability, by the union bound. Using the fact that  $2k < n$ , we get that  $\Pr(E_x) \leq \Pr(\bar{F}_x) < 1/n^7$ . ◀



Finally, we show that even if we knew the vectors  $v_1, \dots, v_{k-1}$ , we cannot guess a vector  $x$  that is an  $\epsilon$ -approximate solution to our problem, because it won't have enough overlap with  $v_k$ , which is unknown. In other words, for an algorithm to output an  $\epsilon$ -optimal solution, it essentially must know the entire set  $V$ .

► **Lemma 15** (Cannot guess  $x^*$ ). *Let  $k > 0$  be an integer and  $\{v_1, \dots, v_{k-1}\}$  be a set of orthonormal vectors. Let  $v_k$  be chosen uniformly at random from  $\text{span}(v_1, \dots, v_{k-1})^\perp$  and let  $V = (v_1, \dots, v_k)$ . Then*

$$\forall x \in B(\vec{0}, 1) : \Pr_{v_k}(f_V(x) - f_V(x^*) \leq \epsilon) \leq 2e^{-\Omega(k^2)}. \quad (18)$$

**Proof.** From Lemma 12, we know that an  $\epsilon$ -optimal solution  $x$  must satisfy  $\langle v_k, x \rangle \leq -8\epsilon$ . But  $v_k$  is chosen uniformly at random from the space  $\text{span}(v_1, \dots, v_{k-1})^\perp$  and any vector  $x \in B(\vec{0}, 1)$  projected to that space also has length at most 1. So from Proposition 13 we know that for any  $x \in B(\vec{0}, 1)$ ,

$$\Pr_{v_k}(\langle v_k, x \rangle \leq -8\epsilon) \leq \Pr_{v_k}(|\langle v_k, x \rangle| \geq 8\epsilon) \leq 2e^{-32(n-k+1)\epsilon^2} \leq 2e^{-\Omega(k^2)}. \quad (19)$$

### 4.3 Quantum query model

We now formally define the quantum query model in our setting. In the usual quantum query model the set of allowed queries is finite, whereas in our setting it is natural to allow the quantum algorithm to query the oracles at any point  $x \in \mathbb{R}^n$ . Due to Lemma 11, it is sufficient to allow the algorithm to query any  $x \in B(\vec{0}, 1)$ , but this is still a continuous space of queries, and hence a query vector could be a superposition over infinitely many states. Instead of formalizing this notion of quantum algorithms, we allow the algorithm to make discrete queries only, but to arbitrarily high precision. The reader is encouraged to not get bogged down by details and to think of the registers as storing the real values that they ideally should, but in the rest of this section we define these algorithms more carefully so that all the spaces involved are finite and well defined. This formalization is not specific to the quantum setting and is done classically as well if we do not want to manipulate real numbers as atomic objects.

All the real numbers that appear will be represented using some  $b$  bits of precision, where  $b$  can be chosen by the algorithm. The reader should imagine  $b$  being arbitrarily large, say exponentially larger than all the parameters involved in the problem, so that the inaccuracy involved by using this representation is negligible. Then the algorithm represents the input  $x \in B(\vec{0}, 1)$  using  $b$  bits of precision per coordinate. The oracle's response will also use  $b$  bits of precision per real number. For a given choice of  $b$ , the quantum algorithm will have some probability of success of solving the problem at hand. We then define the success probability of quantum algorithms that make  $q$  queries by taking a supremum over all  $b$  of  $q$ -query algorithms that solve the problem.

We can now define the oracles more precisely. Classically, the function oracle for a function  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  would simply implement the map  $x \mapsto f(x)$ , where we represent each entry of  $x$  and the output  $f(x)$  using  $b$  bits, so  $x \in \{0, 1\}^{bn}$  and  $f(x) \in \{0, 1\}^b$ . Let's say we have a classical circuit that implements this map using  $G$  gates, say over the gate set of AND, OR, and NOT gates. Then it is easy to construct, in a completely black-box way, a quantum circuit using  $O(G)$  gates (say over the gate set of Hadamard, CNOT, and T) that performs the unitary  $U|x\rangle|y\rangle = |x\rangle|y \oplus f(x)\rangle$ , for every  $x \in \{0, 1\}^{bn}$ , and  $y \in \{0, 1\}^b$ . This is why it is standard to assume that the quantum oracle corresponding to the classical map  $x \mapsto f(x)$  is a unitary that performs  $U|x\rangle|y\rangle = |x\rangle|y \oplus f(x)\rangle$ . We apply the same construction for



the  $\mathcal{FO}(f)$  oracle to get the quantum analogue of the classical map  $x \mapsto g_x$ , where  $g_x$  is some subgradient of  $f$  at  $x$ . Lastly, for convenience we will combine both the function and subgradient oracle into one oracle that when queried with  $x$  returns  $f(x)$  and a subgradient at  $x$ . Since our function family is parameterized by  $V \in \mathcal{V}$ , we call this oracle  $O_V$ .

Let  $\mathcal{A}$  be a quantum query algorithm that makes  $q$  queries.  $\mathcal{A}$  is described by a sequence of unitaries  $U_q O_V U_{q-1} O_V U_{q-2} O_V \cdots U_1 O_V U_0$  applied to an initial state, say  $|0\rangle$ . We assume that the output of  $\mathcal{A}$ , which is a vector  $x$ , is determined by measuring the first  $n$  registers storing real numbers using  $b$  bits.

#### 4.4 Lower bound

We can now prove the quantum lower bound. Let  $\mathcal{A}$  be a  $k-1$  query quantum algorithm that solves Problem 1 on all the functions  $f_V$  for  $V \in \mathcal{V}$ . Due to Lemma 11, we can assume that the algorithm only queries the oracles with vectors  $x \in B(\vec{0}, 1)$ . We also need to describe the behavior of the subgradient oracle on inputs where the subgradient is not unique. On such inputs  $x$ , the subgradient is not unique because several indices  $i \in [k]$  simultaneously achieve the maximum in  $f_V(x)$ . In this case, the subgradient will answer as if the smallest index  $i$  in this set achieved the maximum. Now let  $\mathcal{A}$  be described by the sequence of unitaries  $U_{k-1} O_V U_{k-2} O_V \cdots O_V U_1 O_V U_0$  acting on the starting state  $|0\rangle$ . Let this sequence of unitaries be called  $A$ . Then the final state of the algorithm is  $A|0\rangle$ .

Recall that we defined  $f_V(x) = \max_{i \in [k]} \{g_V^{(i)}(x)\}$ . Let us also define functions  $f_V^{(j)}$  where the maximization is only over the first  $j$  indices instead of all  $k$  indices. Specifically, let  $f_V^{(j)} := \max_{i \in [j]} \{g_V^{(i)}(x)\}$ . We previously defined the oracle  $O_V$  as corresponding to the function  $f_V$ . Let  $O_V^{(j)}$  be the oracle corresponding to the functions  $f_V^{(j)}$ .

Now we define a sequence of unitaries starting with  $A_0 = A$  as follows:

$$\begin{aligned} A_0 &:= U_{k-1} O_V U_{k-2} O_V \cdots O_V U_1 O_V U_0 \\ A_1 &:= U_{k-1} O_V U_{k-2} O_V \cdots O_V U_1 O_V^{(1)} U_0 \\ A_2 &:= U_{k-1} O_V U_{k-2} O_V \cdots O_V^{(2)} U_1 O_V^{(1)} U_0 \\ &\vdots \\ A_{k-1} &:= U_{k-1} O_V^{(k-1)} U_{k-2} O_V^{(k-2)} \cdots O_V^{(2)} U_1 O_V^{(1)} U_0 \end{aligned} \tag{20}$$

We want to show that the algorithm  $A_0$  does not solve our problem. To do so, we will employ the hybrid argument, in which we show that the output of the algorithm  $A_i$  and  $A_{i+1}$  is close, and thus the output of  $A_0$  and  $A_{k-1}$  is close. Finally, we argue that the algorithm  $A_{k-1}$  does not solve our problem because the oracles in the algorithm do not know  $v_k$ . Let us first establish these two claims.

► **Lemma 16** ( $A_{k-1}$  does not solve the problem). *Let  $\mathcal{A}$  be a  $k-1$  query algorithm and let  $A_{k-1}$  be defined as above. Let  $p_V$  be the probability distribution over  $x \in B(\vec{0}, 1)$  obtained by measuring the output state  $A_{k-1}|0\rangle$ . Then  $\Pr_{V, x \sim p_V} (f_V(x) - f_V(x^*) \leq \epsilon) \leq 2e^{-\Omega(k^2)}$ .*

**Proof.** We want to show that the probability (over the random choice of  $V$  and the internal randomness of the algorithm) that  $A_{k-1}$  outputs an  $x$  that satisfies  $f(x) - f(x^*) \leq \epsilon$  is very small.

Let us establish the claim for any fixed choice of  $v_1, \dots, v_{k-1}$ , since if the claim holds for any fixed choice of these vectors, then it also holds for any probability distribution over them. For a fixed choice of vectors, this claim is just  $\Pr_{v_k, x \sim p_V} (f_V(x) - f_V(x^*) \leq \epsilon) \leq 2e^{-\Omega(k^2)}$ .

Now since the algorithm  $A_{k-1}$  only has oracles  $O_V^{(i)}$  for  $i < k$ , the probability distribution  $p_V$  only depends on  $v_1, \dots, v_{k-1}$ . Since these are fixed, this is just a fixed distribution  $p$ . So we can instead establish our claim for all  $x \in B(\vec{0}, 1)$ , which will also establish it for any distribution.

So what we need to establish is that for any  $x \in B(\vec{0}, 1)$ ,  $\Pr_{v_k}(f_V(x) - f_V(x^*) \leq \epsilon) \leq 2e^{-\Omega(k^2)}$ , which is exactly what we showed in Lemma 15.  $\blacktriangleleft$

► **Lemma 17** ( $A_t$  and  $A_{t-1}$  have similar outputs). *Let  $\mathcal{A}$  be a  $k-1$  query algorithm and let  $A_t$  for  $t \in [k-1]$  be the unitaries defined in Equation (20). Then*

$$\mathbb{E}_V(\|A_t|0\rangle - A_{t-1}|0\rangle\|^2) \leq \frac{4}{n^7}. \quad (21)$$

**Proof.** From the definition of the unitaries in Equation (20) and the unitary invariance of the spectral norm, we see that  $\|A_t|0\rangle - A_{t-1}|0\rangle\| = \|(O_V^{(t)} - O_V)U_{t-1}O_V^{(t-1)} \dots O_V^{(1)}U_0|0\rangle\|$ . Let us again prove the claim for any fixed choice of vectors  $v_1, \dots, v_{t-1}$ , which will imply the claim for any distribution over those vectors. Once we have fixed these vectors, the state  $U_{t-1}O_V^{(t-1)} \dots O_V^{(1)}U_0|0\rangle$  is a fixed state, which we can call  $|\psi\rangle$ . Thus our problem reduces to showing for all quantum states  $|\psi\rangle$ ,

$$\mathbb{E}_{v_t, \dots, v_k}(\|(O_V^{(t)} - O_V)|\psi\rangle\|^2) \leq \frac{4}{n^7}. \quad (22)$$

Now we can write an arbitrary quantum state as  $|\psi\rangle = \sum_x \alpha_x |x\rangle |\phi_x\rangle$ , where  $x$  is the query made to the oracle, and  $\sum_x |\alpha_x|^2 = 1$ . Thus the LHS of Equation (22) is equal to

$$\mathbb{E}_{v_t, \dots, v_k} \left( \left\| \sum_x \alpha_x (O_V^{(t)} - O_V) |x\rangle |\phi_x\rangle \right\|^2 \right) \leq \sum_x |\alpha_x|^2 \mathbb{E}_{v_t, \dots, v_k} \left( \|(O_V^{(t)} - O_V)|x\rangle |\phi_x\rangle\|^2 \right). \quad (23)$$

Since  $|\alpha_x|^2$  defines a probability distribution over  $x$ , we can again upper bound the right hand side for any  $x$  instead. Since  $O_V^{(t)}$  and  $O_V$  behave identically for some inputs  $x$ , the only nonzero terms are those where the oracles respond differently, which can only happen if  $f_V^{(t)}(x) \neq f_V(x)$ . When the response is different, we can upper bound  $\|(O_V^{(t)} - O_V)|x\rangle |\phi_x\rangle\|^2$  by 4 using the triangle inequality. Thus for any  $x \in B(\vec{0}, 1)$ , we have

$$\mathbb{E}_{v_t, \dots, v_k} \left( \|(O_V^{(t)} - O_V)|x\rangle |\phi_x\rangle\|^2 \right) \leq 4 \Pr_{v_t, \dots, v_k} (f_V^{(t)}(x) \neq f_V(x)) \leq 4/n^7, \quad (24)$$

where the second inequality follows from Lemma 14. The first inequality requires more explanation. It is based on the claim that if  $f_V^{(t)}(x) = f_V(x)$  then on such inputs  $x$ , the oracles  $O_V^{(t)}$  and  $O_V$  must behave identically. In other words, for such an input  $x$ , for any  $|\phi_x\rangle$ , we have  $O_V^{(t)}|x\rangle |\phi_x\rangle = O_V|x\rangle |\phi_x\rangle$ . The oracle responds with the function value and the subgradient, and we have already assumed that the function values are equal, so we only need to show that if  $f_V^{(t)}(x) = f_V(x)$ , then the subgradient oracle's response is also identical for both functions. Since  $f_V^{(t)}(x) = f_V(x)$ , the maximum in the definition of  $f_V$  was achieved by some index in the first  $t$  indices (and possibly other indices as well). But our definition of the subgradient oracle says that if multiple indices achieve the maximum, the subgradient oracle will respond as if the smallest index achieved the maximum. Hence the response of the subgradient oracle for the two functions will also be identical.  $\blacktriangleleft$

Finally we can put these two lemmas together to prove our lower bound.

► **Lemma 18** ( $\mathcal{A}$  does not solve the problem). *Let  $\mathcal{A}$  be a  $k - 1$  query algorithm. Let  $p_V$  be the probability distribution over  $x \in B(\vec{0}, 1)$  obtained by measuring the output state  $A|0\rangle$ . Then  $\Pr_{V, x \sim p_V}(f_V(x) - f_V(x^*) \leq \epsilon) \leq \frac{1}{\text{poly}(n)}$ .*

**Proof.** Let  $P_V$  be the projection operator that projects a quantum state  $|\psi\rangle$  onto the space spanned by vectors  $|x\rangle$  for  $x$  such that  $f_V(x) - f_V(x^*) \leq \epsilon$ . Then  $\|P_V A|0\rangle\|^2 = \Pr_{x \sim p_V}(f_V(x) - f_V(x^*) \leq \epsilon)$ . We know from Lemma 16 that  $\mathbb{E}_V(\|P_V A_{k-1}|0\rangle\|^2) \leq 2e^{-\Omega(k^2)}$ . We prove our upper bound on the probability by showing that it is approximately the same as  $\mathbb{E}_V(\|P_V A_{k-1}|0\rangle\|^2)$ .

Lemma 17 states that for all  $1 \leq t < k$ ,  $\mathbb{E}_V(\|A_t|0\rangle - A_{t-1}|0\rangle\|^2) \leq \frac{4}{n^7}$ . Using telescoping sums and the Cauchy-Schwarz inequality, we see that

$$\mathbb{E}_V(\|A_{k-1}|0\rangle - A|0\rangle\|^2) \leq \mathbb{E}_V\left(\left(\sum_{t \in [k-1]} \|A_t|0\rangle - A_{t-1}|0\rangle\|\right)^2\right) \quad (25)$$

$$\leq \mathbb{E}_V\left(\sum_{t \in [k-1]} \|A_t|0\rangle - A_{t-1}|0\rangle\|^2\right) \left(\sum_{t \in [k-1]} 1^2\right) \leq \frac{4k}{n^7} \cdot k. \quad (26)$$

For all  $V$ ,

$$\| \|P_V A_{k-1}|0\rangle\| - \|P_V A|0\rangle\| \| \leq \|P_V A_{k-1}|0\rangle - P_V A|0\rangle\| \leq \|A_{k-1}|0\rangle - A|0\rangle\| \quad (27)$$

and so

$$\mathbb{E}_V(\left(\|P_V A_{k-1}|0\rangle\| - \|P_V A|0\rangle\|\right)^2) \leq \frac{4k^2}{n^7}. \quad (28)$$

We want an upper bound on  $\mathbb{E}_V(\|P_V A|0\rangle\|^2 - \|P_V A_{k-1}|0\rangle\|^2)$ , which is no larger than  $2\mathbb{E}_V(\|P_V A|0\rangle\| - \|P_V A_{k-1}|0\rangle\|)$  since  $\|P_V A|0\rangle\| + \|P_V A_{k-1}|0\rangle\| \leq 2$ . We get such a bound by applying Jensen's inequality to Equation (28):  $\mathbb{E}_V(\|P_V A|0\rangle\| - \|P_V A_{k-1}|0\rangle\|) \leq 2k/\sqrt{n^7}$ , and so  $\mathbb{E}_V(\|P_V A|0\rangle\|^2 - \|P_V A_{k-1}|0\rangle\|^2) \leq 4k/\sqrt{n^7}$ .

We can now use linearity of expectation and upper bound our required probability as

$$\Pr_{V, x \sim p_V}(f_V(x) - f_V(x^*) \leq \epsilon) = \mathbb{E}_V(\|P_V A|0\rangle\|^2) \leq 2e^{-\Omega(k^2)} + 4k/\sqrt{n^7} \leq \frac{1}{\text{poly}(n)}. \quad (29)$$

Note that this establishes a statement similar to Theorem 5, except with a polynomially larger value of  $n$ . This result is sufficient to establish the optimality of gradient descent in the dimension-independent setting. To reduce the value of  $n$ , we modify the function above using a *wall function* introduced by [12]. The details of this modification and the resulting quantitatively better lower bound can be found in the full version of the paper.

## 5 Open problems

We showed that in the black-box setting, no quantum algorithm can beat gradient descent in general, in the dimension-independent regime. Here are some interesting questions left open by our work:

1. We showed in Theorem 4 that the class of functions used in the randomized lower bound can be solved faster with quantum queries. Is there a more interesting class of functions on which we can achieve a quantum speedup?

2. Can the quantum lower bound in Section 4 be made to work using the simpler class of functions  $f_V(x) = \max_i \langle v_i, x \rangle$ , which is our function with  $\gamma = 0$ ? If so, this might also decrease the dimension  $n$  required.
3. Can we establish tight quantum lower bounds in the parameter regime where dimension-dependent algorithms outperform gradient descent? When  $1/\epsilon$  is a large polynomial in  $n$ , the complexity of gradient descent is also a large polynomial in  $n$ , but a dimension-dependent algorithm such as the center of gravity method [11] yields an  $O(n \log n)$  upper bound. Can we establish an  $\tilde{\Omega}(n)$  lower bound in this regime? The function used in the randomized lower bound in Section 3 yields an  $\tilde{\Omega}(\sqrt{n})$  lower bound and this is the best bound we are aware of. This is essentially the same as the problem left open by [13, 3], but phrased in the language of membership and separation oracles.
4. What can we say about other standard settings in convex optimization beyond first-order non-smooth convex optimization? Other natural settings include assuming the function is smooth, having the ability to query a prox oracle instead of a subgradient oracle, etc. Can quantum algorithms provide a speedup in the black-box model in these settings over the respective best classical algorithms in that setting?

---

## References

- 1 Joran van Apeldoorn and András Gilyén. Improvements in Quantum SDP-Solving with Applications. In *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*, volume 132 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 99:1–99:15. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2019. doi:10.4230/LIPIcs.ICALP.2019.99.
- 2 Joran van Apeldoorn, András Gilyén, Sander Gribling, and Ronald de Wolf. Quantum SDP-solvers: Better upper and lower bounds. In *58th Annual Symposium on Foundations of Computer Science (FOCS 2017)*, October 2017. doi:10.1109/focs.2017.44.
- 3 Joran van Apeldoorn, András Gilyén, Sander Gribling, and Ronald de Wolf. Convex optimization using quantum oracles. *Quantum*, 4:220, January 2020. doi:10.22331/q-2020-01-13-220.
- 4 Eric Balkanski and Yaron Singer. Parallelization does not accelerate convex optimization: Adaptivity lower bounds for non-smooth convex minimization. *arXiv preprint*, 2018. arXiv:1808.03880.
- 5 Keith Ball. An elementary introduction to modern convex geometry. In Silvio Levy, editor, *Flavors of geometry*, volume 31, pages 1–58. Cambridge University Press, 1997. URL: <http://library.msri.org/books/Book31/files/ball.pdf>.
- 6 Walter Baur and Volker Strassen. The complexity of partial derivatives. *Theoretical Computer Science*, 22(3):317–330, 1983. doi:10.1016/0304-3975(83)90110-X.
- 7 Aleksandrs Belovs. Quantum algorithms for learning symmetric juntas via adversary bound. In *Proceedings of the 2014 IEEE 29th Conference on Computational Complexity (CCC 2014)*, CCC '14, page 22–31, 2014. doi:10.1109/CCC.2014.11.
- 8 Charles H. Bennett, Ethan Bernstein, Gilles Brassard, and Umesh Vazirani. Strengths and weaknesses of quantum computing. *SIAM Journal on Computing*, 26(5):1510–1523, 1997. doi:10.1137/S0097539796300933.
- 9 Fernando G. S. L. Brandão, Amir Kalev, Tongyang Li, Cedric Yen-Yu Lin, Krysta M. Svore, and Xiaodi Wu. Quantum SDP Solvers: Large Speed-Ups, Optimality, and Applications to Quantum Learning. In *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*, volume 132 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 27:1–27:14. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2019. doi:10.4230/LIPIcs.ICALP.2019.27.

- 10 Fernando G.S.L. Brandão and Krysta M. Svore. Quantum speed-ups for solving semidefinite programs. In *58th Annual Symposium on Foundations of Computer Science (FOCS 2017)*, October 2017. doi:10.1109/focs.2017.45.
- 11 Sébastien Bubeck. Convex optimization: Algorithms and complexity. *Found. Trends Mach. Learn.*, 8(3–4):231–357, November 2015. doi:10.1561/22000000050.
- 12 Sébastien Bubeck, Qijia Jiang, Yin Tat Lee, Yuanzhi Li, and Aaron Sidford. Complexity of highly parallel non-smooth convex optimization. In *Advances in Neural Information Processing Systems 32 (NeurIPS 2019)*, pages 13900–13909, 2019. URL: <http://papers.nips.cc/paper/9541-complexity-of-highly-parallel-non-smooth-convex-optimization>.
- 13 Shouvanik Chakrabarti, Andrew M. Childs, Tongyang Li, and Xiaodi Wu. Quantum algorithms and lower bounds for convex optimization. *Quantum*, 4:221, January 2020. doi:10.22331/q-2020-01-13-221.
- 14 Jelena Diakonikolas and Cristóbal Guzmán. Lower bounds for parallel and randomized convex optimization. In *Proceedings of the Thirty-Second Conference on Learning Theory*, volume 99, pages 1132–1157. PMLR, 2019. URL: <http://proceedings.mlr.press/v99/diakonikolas19c.html>.
- 15 András Gilyén, Srinivasan Arunachalam, and Nathan Wiebe. Optimizing quantum optimization algorithms via faster quantum gradient computation. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA '19*, page 1425–1444, 2019. doi:10.1137/1.9781611975482.87.
- 16 Andreas Griewank and Andrea Walther. *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation, Second Edition*. Other Titles in Applied Mathematics. Society for Industrial and Applied Mathematics, 2008. URL: <https://books.google.com/books?id=xoiLaRxcBEC>.
- 17 Stephen P. Jordan. Fast quantum algorithm for numerical gradient estimation. *Phys. Rev. Lett.*, 95:050501, July 2005. doi:10.1103/PhysRevLett.95.050501.
- 18 Sham M Kakade and Jason D Lee. Provably correct automatic sub-differentiation for qualified programs. In *Advances in Neural Information Processing Systems 31 (NeurIPS 2018)*, pages 7125–7135, 2018. URL: <http://papers.nips.cc/paper/7943-provably-correct-automatic-sub-differentiation-for-qualified-programs>.
- 19 Iordanis Kerenidis and Anupam Prakash. Quantum gradient descent for linear systems and least squares. *Physical Review A*, 101(2):022316, 2020. doi:10.1103/PhysRevA.101.022316.
- 20 Yin Tat Lee, Aaron Sidford, and Santosh S. Vempala. Efficient convex optimization with membership oracles. In Sébastien Bubeck, Vianney Perchet, and Philippe Rigollet, editors, *Conference On Learning Theory, COLT 2018, Stockholm, Sweden, 6-9 July 2018*, volume 75 of *Proceedings of Machine Learning Research*, pages 1292–1294. PMLR, 06–09 July 2018. URL: <http://proceedings.mlr.press/v75/lee18a.html>.
- 21 A. Nemirovski. On parallel complexity of nonsmooth convex optimization. *Journal of Complexity*, 10(4):451–463, 1994. doi:10.1006/jcom.1994.1025.
- 22 Arkadii Nemirovsky and David Borisovich Yudin. *Problem complexity and method efficiency in optimization*. Wiley, 1983.
- 23 Yurii Nesterov. *Introductory Lectures on Convex Optimization*. Springer US, 2004. doi:10.1007/978-1-4419-8853-9.
- 24 Yurii Nesterov. *Lectures on convex optimization*, volume 137. Springer, 2018. doi:10.1007/978-3-319-91578-4.
- 25 Patrick Rebentrost, Maria Schuld, Leonard Wossnig, Francesco Petruccione, and Seth Lloyd. Quantum gradient descent and Newton’s method for constrained polynomial optimization. *New Journal of Physics*, 21(7):073023, 2019. doi:10.1088/1367-2630/ab2a9e.
- 26 Blake Woodworth and Nathan Srebro. Lower bound for randomized first order convex optimization. *arXiv preprint*, 2017. arXiv:1709.03594.

# Quantum Versus Randomized Communication Complexity, with Efficient Players

**Uma Girish**

Department of Computer Science, Princeton University, NJ, USA  
ugirish@cs.princeton.edu

**Ran Raz**

Department of Computer Science, Princeton University, NJ, USA  
ranr@cs.princeton.edu

**Avishay Tal**

Department of Electrical Engineering and Computer Sciences,  
University of California at Berkeley, CA, USA  
atal@berkeley.edu

---

## Abstract

We study a new type of separations between quantum and classical communication complexity, separations that are obtained using quantum protocols where all parties are **efficient**, in the sense that they can be implemented by small quantum circuits, with oracle access to their inputs. Our main result qualitatively matches the strongest known separation between quantum and classical communication complexity [8] and is obtained using a quantum protocol where all parties are efficient. More precisely, we give an explicit partial Boolean function  $f$  over inputs of length  $N$ , such that:

- (1)  $f$  can be computed by a simultaneous-message quantum protocol with communication complexity  $\text{polylog}(N)$  (where at the beginning of the protocol Alice and Bob also have  $\text{polylog}(N)$  entangled EPR pairs).
- (2) Any classical randomized protocol for  $f$ , with any number of rounds, has communication complexity at least  $\tilde{\Omega}(N^{1/4})$ .
- (3) All parties in the quantum protocol of Item (1) (Alice, Bob and the referee) can be implemented by quantum circuits of size  $\text{polylog}(N)$  (where Alice and Bob have oracle access to their inputs).

Items (1), (2) qualitatively match the strongest known separation between quantum and classical communication complexity, proved by Gavinsky [8]. Item (3) is new. (Our result is incomparable to the one of Gavinsky. While he obtained a quantitatively better lower bound of  $\Omega(N^{1/2})$  in the classical case, the referee in his quantum protocol is inefficient).

Exponential separations of quantum and classical communication complexity have been studied in numerous previous works, but to the best of our knowledge the efficiency of the parties in the quantum protocol has not been addressed, and in most previous separations the quantum parties seem to be inefficient. The only separations that we know of that have efficient quantum parties are the recent separations that are based on lifting [10, 5]. However, these separations seem to require quantum protocols with at least two rounds of communication, so they imply a separation of two-way quantum and classical communication complexity but they do not give the stronger separations of simultaneous-message quantum communication complexity vs. two-way classical communication complexity (or even one-way quantum communication complexity vs. two-way classical communication complexity).

Our proof technique is completely new, in the context of communication complexity, and is based on techniques from [15]. Our function  $f$  is based on a lift of the FORRELATION problem, using XOR as a gadget.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Communication complexity; Theory of computation  $\rightarrow$  Quantum complexity theory

**Keywords and phrases** Exponential Separation, Quantum, Randomized, Communication, Complexity, Forrelation

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.54



© Uma Girish, Ran Raz, and Avishay Tal;

licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 54; pp. 54:1–54:20

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



**Related Version** A full version of the paper is available at <https://arxiv.org/abs/1911.02218>.

**Funding** *Uma Girish*: Research supported by the Simons Collaboration on Algorithms and Geometry, by a Simons Investigator Award and by the National Science Foundation grant No. CCF-1714779.

*Ran Raz*: Research supported by the Simons Collaboration on Algorithms and Geometry, by a Simons Investigator Award and by the National Science Foundation grant No. CCF-1714779.

*Avishay Tal*: Part of this work was done when the author was a postdoc at the Department of Computer Science, Stanford University. Partially supported by a Motwani Postdoctoral Fellowship and by NSF grant CCF-1763311.

## 1 Introduction

Exponential separations between quantum and classical communication complexity have been established in various models and settings. These separations give explicit examples of partial functions that can be computed by quantum protocols with very small communication complexity, while any classical randomized protocol requires significantly higher communication complexity. However, to the best of our knowledge, in all these works the efficiency of the quantum players in the quantum protocol has not been addressed and in most of these separations, the quantum players are inefficient.

Communication complexity studies the amount of communication needed to perform computational tasks that depend on two (or more) inputs, each given to a different player. The efficiency of the players in a communication complexity protocol is usually not addressed. If the players need to read their entire inputs, their time complexity is at least the length of the inputs. However, the inputs may be represented compactly by a black box and (particularly in the quantum case) we can hope for players that can be implemented very efficiently by small (say, poly-logarithmic size) quantum circuits, with oracle access to their inputs.

Our main result qualitatively matches the strongest known separation between quantum and classical communication complexity [8] and is obtained using quantum protocols where all players are efficient. To prove our results we use a completely different set of techniques, based on techniques from the recent oracle separation of BQP and PH [15].

### 1.1 Previous Work

The relative power of quantum and classical communication complexity has been studied in numerous of works. While it is unknown whether quantum communication can offer exponential advantage over randomized communication for total functions, a series of works gave explicit examples of partial Boolean functions (promise problems) that have quantum protocols with very small communication complexity, while any classical protocol requires exponentially higher communication complexity. The history of exponential advantage of quantum communication, that is most relevant to our work, is briefly summarized below.

Buhrman, Cleve and Wigderson gave the first (exponential) separation between zero-error quantum communication complexity and classical deterministic communication complexity [4]. Raz gave the first exponential separation between two-way quantum communication complexity and two-way randomized communication complexity [14]. Bar-Yossef et al [3] (for search problems) and Gavinsky et al [9] (for promise problems) gave the first (exponential) separations between one-way quantum communication complexity and one-way randomized communication complexity. Klartag and Regev gave the first (exponential) separation between one-way quantum communication complexity and two-way random-



ized communication complexity [16]. Finally, Gavinsky gave an (exponential) separation between simultaneous-message quantum communication complexity and two-way randomized communication complexity [8].

We note that Gavinsky's work is the strongest separation known today and essentially subsumes the separations discussed above. More precisely, Gavinsky [8] gave an explicit partial Boolean function  $f$  over inputs of length  $N$ , such that:

1.  $f$  can be computed by a simultaneous-message quantum protocol with communication complexity  $\text{polylog}(N)$ : Alice and Bob simultaneously send quantum messages of length  $\text{polylog}(N)$  to a referee, who performs a quantum measurement on the messages and announces the answer. (At the beginning of the protocol Alice and Bob also have  $\text{polylog}(N)$  entangled EPR pairs).

We note that this also implies a one-way quantum protocol where Alice sends a message of length  $\text{polylog}(N)$  qubits to Bob, who performs a measurement and announces the answer (or vice versa).

2. Any classical randomized protocol for  $f$  has communication complexity at least  $\Omega(N^{1/2})$ .

A drawback of Gavinsky's separation, in the context of our work, is that the referee in his quantum protocol is inefficient as it is required to perform  $O(N)$  quantum operations (and this seems to be crucial in his lower bound proof).

As mentioned before, to the best of our knowledge, the efficiency of the quantum players has not been addressed in previous works on separations of quantum and classical communication complexity. The only separations that we know of that do have efficient quantum parties are the separations that follow from the recent randomized query-to-communication lifting theorems of [10, 5], applied to problems for which we know that quantum decision trees offer an exponential advantage over randomized ones, such as the FORRELATION problem of [1, 2]. However, lifting with the gadgets used in [10, 5] seems to require quantum protocols with two rounds of communication. Thus, these theorems only imply a separation of two-way quantum and classical communication complexity and do not give the stronger separations of simultaneous-message quantum communication complexity vs. two-way classical communication complexity (or even one-way quantum communication complexity vs. two-way classical communication complexity).

## 1.2 Our Result

We recover Gavinsky's state of the art separation, using entirely different techniques. While the parameters in our bounds are weaker, our quantum protocol is *efficient*, in the sense that it involves just  $\text{polylog}(N)$  amount of work by Alice, Bob and the referee, when the players have blackbox access to their inputs. In other words, the output of the entire simultaneous protocol can be described by a  $\text{polylog}(N)$  size quantum circuit, with oracle access to the inputs.

More precisely, our main result gives an explicit partial Boolean function  $f$  over inputs of length  $N$ , such that:

1. As in Gavinsky's work,  $f$  can be computed by a simultaneous-message quantum protocol with communication complexity  $\text{polylog}(N)$ : Alice and Bob simultaneously send quantum messages of length  $\text{polylog}(N)$  to a referee, who performs a quantum measurement on the messages and announces the answer. (At the beginning of the protocol Alice and Bob also have  $\text{polylog}(N)$  entangled EPR pairs).

As before, this also implies a one-way quantum protocol where Alice sends a message of length  $\text{polylog}(N)$  qubits to Bob, who performs a measurement and announces the answer (or vice versa).

2. Any classical randomized protocol for  $f$  has communication complexity at least  $\tilde{\Omega}(N^{1/4})$ .
3. All parties in the quantum protocol of Item (1) (Alice, Bob and the referee) can be implemented by quantum circuits of size  $\text{polylog}(N)$  (where Alice and Bob have oracle access to their input).

The problem that we define is a lift of the FORRELATION problem of [1, 2, 15] with XOR as the gadget. Our proof technique follows the Fourier-analysis framework of [15]. Our proof offers an entirely new and possibly simpler approach for communication complexity lower bounds. We believe this technique may be applicable in a broader setting. We note that lower bounds for lifting by XOR, using a Fourier-analysis approach, were previously studied in [13, 11].

### 1.3 Our Communication Complexity Problem

Let  $N = 2^n$  and  $H_N$  be the  $N \times N$  normalized Hadamard matrix. Let  $x = (x_1, x_2)$  be an input where  $x_1, x_2 \in \{-1, 1\}^N$ . We define the forrelation of  $x$  as the correlation between the second half  $x_2$  and the Hadamard transform of the first half  $x_1$ .

$$\text{forr}(x) := \left\langle \frac{1}{\sqrt{N}} H_N(x_1) \middle| \frac{1}{\sqrt{N}} x_2 \right\rangle$$

The communication problem for which our separation holds is a lift of the forrelation problem of [15], with XOR as the gadget. Let  $x, y \in \{-1, 1\}^{2N}$ . Alice gets  $x$  and Bob gets  $y$  and their goal is to compute the partial function  $F$  defined by

$$F(x, y) := \begin{cases} 1 & \text{if } \text{forr}(x \cdot y) \geq \frac{1}{200} \cdot \frac{1}{\ln N} \\ -1 & \text{if } \text{forr}(x \cdot y) \leq \frac{1}{400} \cdot \frac{1}{\ln N}. \end{cases}$$

Here  $x \cdot y$  refers to the coordinate-wise product of the vectors  $x, y$ . We refer to this problem as the forrelation problem.

► **Theorem 1.** *The forrelation problem can be solved in the quantum simultaneous with entanglement model with  $O(\log^3 N)$  bits of communication, when Alice and Bob are given access to  $O(\log^3 N)$  bits of shared entanglement. Moreover, the protocol is efficient, as it can be implemented by a  $O(\log^3 N)$  size quantum circuit with oracle access to inputs.*

The quantum upper bound on  $F$  follows from the fact that the XOR of the inputs can be computed by a simultaneous-message quantum protocol, when the players share entanglement, and the fact that  $\text{forr}(x)$  can be estimated by a small size quantum circuit [1, 2, 15].

► **Theorem 2.** *The randomized bounded-error interactive communication cost of the forrelation problem is  $\tilde{\Omega}(N^{\frac{1}{4}})$ .*

### 1.4 An Overview of the Lower Bound

In this section, we outline the proof of the lower bound. We use the forrelation distribution  $\mathcal{D}$  on  $\{-1, 1\}^{2N}$  as defined by [15]. We define a distribution  $\mathcal{V}$  on inputs to the communication problem, obtained by sampling  $z \sim \mathcal{D}$ , and  $x \in \{-1, 1\}^{2N}$  uniformly at random, and setting  $y := x \cdot z$ . Alice gets  $x$  and Bob gets  $y$ . It can be shown that the distribution  $\mathcal{V}$  has considerable support over the yes instances of  $F$ , while the uniform distribution  $\mathcal{U}$  on  $\{-1, 1\}^{4N}$  has large support over the no instances of  $F$ . This fact along with the following theorem implies a lower bound on the randomized communication cost of  $F$ .

► **Theorem 3.** *Consider the following distribution. A string  $z \in \{-1, 1\}^{2N}$  is drawn from the forrelation distribution,  $x \sim U_{2N}$  is drawn uniformly and  $y := x \cdot z$ . Alice gets  $x$  and Bob gets  $y$ . Given any deterministic communication protocol  $C : \{-1, 1\}^{2N} \times \{-1, 1\}^{2N} \rightarrow \{-1, 1\}$  of cost  $c \geq 1$ , its expectation when the inputs are drawn from this distribution is close to when the inputs are drawn from the uniform distribution. That is,*

$$\left| \mathbb{E}_{\substack{x \sim U_{2N} \\ z \sim \mathcal{D}}} [C(x, x \cdot z)] - \mathbb{E}_{x, y \sim U_{2N}} [C(x, y)] \right| \leq O\left(\frac{c^2}{N^{1/2}}\right).$$

*In other words, no deterministic protocol of cost  $o(N^{1/4})$  has considerable advantage in distinguishing the above distribution from the uniform distribution.*

We now outline the proof of this theorem. Any cost  $c$  protocol induces a partition of the input space into at most  $2^c$  rectangles. Let  $A \times B$  be any rectangle, and let  $\mathbb{1}_A, \mathbb{1}_B : \{-1, 1\}^{2N} \rightarrow \{0, 1\}$  be the indicator functions of  $A$  and  $B$  respectively. Note that for all distributions  $\mathcal{S}$  on  $\{-1, 1\}^{2N}$ , we have

$$\mathbb{E}_{z \sim \mathcal{S}} \mathbb{E}_{x \sim U_{2N}} [\mathbb{1}_A(x) \mathbb{1}_B(x \cdot z)] = \mathbb{E}_{z \sim \mathcal{S}} [(\mathbb{1}_A * \mathbb{1}_B)(z)].$$

Here, the notation  $f * g$  refers to the convolution of Boolean functions  $f$  and  $g$ . This identity implies that our goal is to show that the expectation of the function  $\sum_{A \times B} (\mathbb{1}_A * \mathbb{1}_B)(z)$  over a uniformly distributed  $z$  is close to the expectation over  $z \sim \mathcal{D}$ . An essential contribution of the works of [15] and [7] is the following result. For any family of functions  $\mathcal{F}$  that is closed under restrictions, to show that the family is fooled by the forrelation distribution, it suffices to bound the  $\ell_1$ -norm of the second level Fourier coefficients of the family. More precisely, the maximum advantage of a function  $f \in \mathcal{F}$  in distinguishing the uniform distribution and  $\mathcal{D}$ , is at most  $O\left(\frac{1}{\sqrt{N}}\right)$  times the maximum second level Fourier mass of a function  $f \in \mathcal{F}$ . Since small cost communication protocols form a family of functions closed under restrictions, the same reasoning applies here. In this paper however, we present a complete proof of this connection. We then provide the following bound on the second level Fourier mass corresponding to a small cost protocol.

▷ **Claim 1.** Let  $C(x, y) : \{-1, 1\}^{2N} \times \{-1, 1\}^{2N} \rightarrow \{-1, 1\}$  be any deterministic protocol of cost  $c \geq 1$ , let  $D(x, z) : \mathbb{R}^{2N} \times \mathbb{R}^{2N} \rightarrow \mathbb{R}$  refer to the unique multilinear extension of  $C(x, x \cdot z)$  and  $H : \mathbb{R}^{2N} \rightarrow \mathbb{R}$  be defined by  $H(z) = \mathbb{E}_{x \sim U_{2N}} D(x, z)$ . Then,

$$L_2(H) \triangleq \sum_{|S|=2} |\widehat{H}(S)| \leq 120c^2.$$

We now describe the proof of this claim. Let  $A \times B$  be a rectangle in the partition induced by the cost  $c$  protocol. An important property of the convolution of two functions  $f, g$  is that for all subsets  $S \subseteq [n]$ , we have  $\widehat{f * g}(S) = \widehat{f}(S) \widehat{g}(S)$ . This, along with Cauchy-Schwarz implies that

$$\sum_{|S|=2} |\widehat{\mathbb{1}_A * \mathbb{1}_B}(S)| = \sum_{|S|=2} |\widehat{\mathbb{1}_A}(S) \widehat{\mathbb{1}_B}(S)| \leq \left( \sum_{|S|=2} \widehat{\mathbb{1}_A}(S)^2 \right)^{1/2} \left( \sum_{|S|=2} \widehat{\mathbb{1}_B}(S)^2 \right)^{1/2}.$$

We then use a well known inequality on Fourier coefficients. It appears as “Level-k Inequalities” in Ryan Odonnell’s book [12, Chapter 9.5] and it states that for a function  $f : \{-1, 1\}^n \rightarrow \{0, 1\}$  with expectation  $\mathbb{E}[f] = \alpha$ , for any  $k \leq 2 \ln(1/\alpha)$ , we have  $\sum_{|S|=k} (\widehat{f}(S))^2 \leq$

$O(\alpha^2 \ln^k(1/\alpha))$ . For simplicity, assume that  $|A| = |B| = 2^{(n-c)/2}$ . The previous paragraphs and the assumption that  $\mathbb{E}[\mathbb{1}_A], \mathbb{E}[\mathbb{1}_B] = \frac{1}{2^{c/2}}$  imply that the advantage of a rectangle is at most  $O\left(\frac{1}{\sqrt{N}} \frac{1}{2^c} c^2\right)$ . Adding the contributions from all rectangles implies that the advantage of a cost  $c$  protocol is at most  $O\left(\frac{c^2}{\sqrt{N}}\right)$ . This implies that every protocol of cost  $o(N^{1/4})$  has advantage at most  $o(1)$  in distinguishing between  $\mathcal{U}$  and  $\mathcal{V}$ . The bound in the case of a general partition follows from a concavity argument. This completes the proof overview.

### Open Questions

We conjecture that the correct randomized communication complexity for this problem is  $\tilde{\Omega}(\sqrt{N})$  and that the above proof technique can be strengthened to show this. One way to do this would be to show a better bound on the Fourier coefficients of deterministic communication protocols. In particular, it would suffice to show a bound of  $O(c \cdot \text{poly} \log(N))$  on the second level Fourier mass of protocols with  $c$ -bits of communication.

## 2 Preliminaries

For  $n \in \mathbb{N}$ , let  $[n]$  denote the set  $\{1, 2, \dots, n\}$ . For a vector  $x \in \mathbb{R}^n$  and  $i \in [n]$ , we refer to the  $i$ -th coordinate of  $x$  by either  $x(i)$  or  $x_i$ . For a subset  $S \subset [n]$ , let  $x_S \in \mathbb{R}^{|S|}$  be the restriction of  $x$  to coordinates in  $S$ . For vectors  $x, y \in \mathbb{R}^n$ , let  $x \cdot y$  be their point-wise product, i.e., the vector whose  $i$ -th coordinate is  $x_i y_i$ . Let  $\langle x | y \rangle$  be the real inner product  $\sum_i x_i y_i$  between  $x$  and  $y$ . Let  $v^{-1}$  be the coordinate-wise inverse of a vector  $v \in (\mathbb{R} \setminus 0)^n$ .

### 2.1 Fourier Analysis on the Boolean Hypercube

The set  $\{-1, 1\}^n$  is referred to as the Boolean hypercube in  $n$  dimensions, or the  $n$ -dimensional hypercube. We sometimes refer to it by  $\{0, 1\}^n$ , using the bijection mapping  $(x_1, \dots, x_n) \in \{0, 1\}^n$  to  $((-1)^{x_1}, \dots, (-1)^{x_n}) \in \{-1, 1\}^n$ . We also represent elements of  $\{-1, 1\}^n$  by elements of  $[2^n]$ , using the bijection mapping  $((-1)^{x_1}, \dots, (-1)^{x_n}) \in \{-1, 1\}^n$  to  $1 + \sum_{i=1}^n 2^{i-1} x_i \in [2^n]$ . We typically use  $N$  to denote  $2^n$ . Let  $\mathbb{I}_n$  denote the  $n \times n$  identity matrix. Let  $U_n$  be the uniform distribution on  $\{-1, 1\}^n$ . Let  $\mathcal{F} := \{f : \{-1, 1\}^n \rightarrow \mathbb{R}\}$  be the set of all functions from the  $n$ -dimensional hypercube to the real numbers. This is a real vector space of dimension  $2^n$ . We define an inner product over this space. For every  $f, g \in \mathcal{F}$ , let

$$\langle f, g \rangle := \mathbb{E}_{x \sim U_n} [f(x)g(x)].$$

For any universe  $\mathcal{U}$  and a subset  $S \subseteq \mathcal{U}$ , we use  $\mathbb{1}_S : \mathcal{U} \rightarrow \{0, 1\}$  to refer to the indicator function of  $S$  defined by:

$$\mathbb{1}_S(x) := \begin{cases} 1 & \text{if } x \in S \\ 0 & \text{otherwise.} \end{cases}$$

The set of indicator functions of singleton sets  $\{\mathbb{1}_{\{a\}} : a \in \{-1, 1\}^n\}$  is the standard orthogonal basis for  $\mathcal{F}$ . The character functions form an orthonormal basis for  $\mathcal{F}$ . These are functions  $\chi_S : \{-1, 1\}^n \rightarrow \{-1, 1\}$  associated to every set  $S \subseteq [n]$  and are defined at every point  $x \in \{-1, 1\}^n$  by  $\chi_S(x) := \prod_{i \in S} x_i$ . For a function  $f \in \mathcal{F}$ , and  $S \subseteq [n]$ , we define its  $S$ -th Fourier coefficient to be  $\hat{f}(S) := \mathbb{E}_{x \sim U_n} [f(x) \chi_S(x)]$ . Every  $f \in \mathcal{F}$  can be expressed as  $f(x) = \sum_{S \subseteq [n]} \hat{f}(S) \chi_S(x)$ . For  $f : \{-1, 1\}^n \rightarrow \mathbb{R}$  and  $k \in \{0, \dots, n\}$ , let  $L_k(f) := \sum_{S \subseteq [n], |S|=k} |\hat{f}(S)|$  refer to the level  $k$  Fourier mass of  $f$ .

Given functions  $f, g : \{-1, 1\}^n \rightarrow \mathbb{R}$ , their convolution  $f * g : \{-1, 1\}^n \rightarrow \mathbb{R}$  is defined as  $f * g(x) := \mathbb{E}_{y \sim U_n} [f(y)g(y \cdot x)]$ . A standard fact about convolution of functions is that  $\widehat{f * g}(S) = \widehat{f}(S)\widehat{g}(S)$  for all  $S \subseteq [n]$ .

## 2.2 Quantum Computation

Let  $\mathcal{H}_m$  be the Hilbert space of dimension  $2^m$  defined by the complex span of the orthonormal basis  $\{|x\rangle : x \in \{-1, 1\}^m\}$ . We sometimes express these basis elements by integers  $\{|i\rangle : i \in [2^m]\}$  by the same correspondence as before.

Fix any universal set of gates for quantum computation. A quantum circuit  $Q : \{-1, 1\}^n \rightarrow \{-1, 1\}^m$  of space  $S$  consists of a set of  $S$  registers, the first  $n$  of which are initialized to  $|x\rangle$ , the input, while the rest are initialized to  $|1\rangle$ . It further consists of a sequence of operators chosen from the universal set of gates, along with a description of which register they act on. The size of a circuit is the number of operators. The output of a circuit is defined to be the contents of the first  $m$  registers. Since we want the output to be Boolean, we assume that the circuit measures these registers and returns the outcome. Thus, a quantum circuit is inherently probabilistic.

We now describe quantum circuits with *query or oracle* access. In this model, all registers are initialized to  $|1\rangle$  and the input  $x \in \{-1, 1\}^n$  is not written into the registers. Instead, it is compactly presented to the algorithm using a blackbox, a device which for every index  $i \in [n]$ , returns  $x(i)|i\rangle$  when it is given  $|i\rangle$  as input. More precisely, for every possible input  $x \in \{-1, 1\}^n$ , the oracle to  $x$  is the linear operator  $O_x : \mathcal{H}_{\lceil \log n \rceil} \rightarrow \mathcal{H}_{\lceil \log n \rceil}$  which maps the basis states  $|i\rangle$  to  $x_i|i\rangle$  whenever  $i \in [n]$  and otherwise leaves it fixed. This indeed restricts to a unitary operation on pure states, as its action on the basis states is described by a diagonal  $\{-1, 1\}$ -matrix. This serves as the quantum analogue of a classical oracle, which is a blackbox that returns  $x(i)$  on input  $i \in [n]$ . A *quantum circuit with oracle access to inputs* is a quantum circuit that is allowed to use the  $O_x$  operator in addition to the usual operators, where  $x$  is the input to the computation. The *size* of the circuit is the total number of gates used from the universal gate set plus the number of oracle queries used. We say that an algorithm is *efficient*, if it is described by a circuit of size at most  $\text{poly} \log n$  with oracle access to inputs. Note that it is possible to use the oracle  $O_x$  to explicitly write down the input  $x$  into  $n$  registers, however, this requires  $n$  oracle calls and  $n$  registers. It is often the case that this step is unnecessary.

## 2.3 Classical & Quantum Communication Complexity

Let  $f : \{-1, 1\}^n \times \{-1, 1\}^m \rightarrow \{-1, 1\}$  be a partial Boolean function. Alice (respectively Bob) receives a private input  $x \in \{-1, 1\}^n$  (respectively  $y \in \{-1, 1\}^m$ ) and the players' goal is to compute  $f(x, y)$  if  $(x, y)$  is in the support of  $f$ , while exchanging as few bits as possible. An input  $(x, y)$  is said to be a YES (respectively NO) instance if  $f(x, y) = -1$  (respectively if  $f(x, y) = 1$ ). We assume familiarity with *bounded-error randomized* and *quantum communication complexity*. In quantum communication with *entanglement*, Alice and Bob are given  $m$  independent copies of the Bell state for some  $m \in \mathbb{N}$ . In this case, we say that Alice and Bob share  $m$  bits of entanglement. In the *simultaneous* model of communication, Alice and Bob are not allowed to exchange messages with each other. Instead, they are allowed one round of communication with a referee Charlie, to whom they can only send qubits. The referee then performs some quantum operation on the qubits he receives and returns a bit as the output. As before, a bounded-error simultaneous protocol computes  $f$  if for all  $(x, y)$  in the support of  $f$ , with probability at least  $2/3$ , the referee's output agrees with  $f(x, y)$ . The cost is the total number of qubits that Alice and Bob send the referee.

Note that in each of the above models of communication, every function  $f : \{-1, 1\}^n \times \{-1, 1\}^m \rightarrow \{-1, 1\}$  has communication cost at most  $n + m$ , since the players may simply reveal their entire inputs. Hence, a small cost protocol is one in which the communication cost is at most  $\text{poly} \log(n + m)$ .

A communication protocol is said to be *efficient* if it can be implemented by a small size circuit with oracle access  $O_x, O_y$  to the inputs  $x, y$ . Protocols with small communication cost are not necessarily efficient, as they may require computationally intensive processing on the messages, or they may require the players to make several probes into their inputs.

## 2.4 The Forrelation Distribution $\mathcal{D}$

Let  $x \sim \mathcal{D}$  refer to a random variable  $x$  distributed according to the probability distribution  $\mathcal{D}$ . We use  $\mathbb{P}_{\mathcal{D}}$  to refer to the probability measure associated with  $\mathcal{D}$  and  $\mathbb{P}_{x \sim \mathcal{D}}(E(x))$  to refer to the probability of event  $E(x)$  when  $x \sim \mathcal{D}$ . For an event  $E(x)$ , we will denote by  $\mathcal{D}|E(x)$  (respectively  $\mathcal{D}|\neg E(x)$ ), the distribution  $\mathcal{D}$  conditioned on the event  $E(x)$  occurring (respectively, the event  $E(x)$  not occurring). Let  $\epsilon \geq 0$  be a parameter,  $f(x) : \mathbb{R}^n \rightarrow \mathbb{R}$  a function and  $\mathcal{D}$  a distribution on  $\mathbb{R}^n$ . We say that  $\mathcal{D}$  fools  $f$  with error  $\epsilon$  if  $\left| \mathbb{E}_{x \sim U_n}[f(x)] - \mathbb{E}_{x \sim \mathcal{D}}[f(x)] \right| \leq \epsilon$ .

Let  $\mathcal{N}(\mu, \sigma^2)$  denote a Gaussian distribution of mean  $\mu \in \mathbb{R}$  and variance  $\sigma^2 \in \mathbb{R}_{\geq 0}$ . We will repeatedly use the following standard facts about Gaussians.

- Gaussian Concentration inequality: For  $X \sim \mathcal{N}(\mu, \sigma^2)$ , we have  $\mathbb{P}[|X - \mu| \geq a] \leq e^{-\frac{a^2}{2\sigma^2}}$ .
- The sum  $\sum_i X_i$  of independent Gaussians  $X_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$  is distributed according to  $\mathcal{N}(\sum_i \mu_i, \sum_i \sigma_i^2)$ .

Let  $N = 2^n$ . The Hadamard matrix  $H_N$  is an  $N \times N$  unitary matrix. We let  $x$  and  $y$  in  $\{0, 1\}^n$  index rows and columns of  $H_N$  respectively. The entries of  $H_N$  are as follows.

$$H_N(x, y) := \begin{cases} \frac{1}{\sqrt{N}} & \text{if } \sum_i x_i y_i \bmod 2 = 0 \\ \frac{-1}{\sqrt{N}} & \text{otherwise} \end{cases}$$

Let  $x = (x_1, x_2)$  for  $x_1, x_2 \in \{-1, 1\}^N$ . We define the forrelation of  $x$  as the correlation between the second half  $x_2$  and the Hadamard transform of the first half  $x_1$ .

$$\text{forr}(x) := \left\langle \frac{1}{\sqrt{N}} H_N(x_1) \middle| \frac{1}{\sqrt{N}} x_2 \right\rangle$$

We state the definition of the forrelation distribution, as defined in [15]. Fix a parameter  $\epsilon = \frac{1}{50 \ln N}$ . We first define an auxilliary Gaussian distribution  $\mathcal{G}$  generated by sampling the first half uniformly at random and letting the second half be the Hadamard transform of the first half. More precisely,

1. Sample  $x_1, \dots, x_N \sim \mathcal{N}(0, \epsilon)$ .
2. Let  $y = H_N x$ .
3. Output  $(x, y)$ .

This is a Gaussian random variable in  $2N$  dimensions of mean 0 and covariance matrix given by

$$\epsilon \begin{bmatrix} \mathbb{I}_N & H_N \\ H_N & \mathbb{I}_N \end{bmatrix}.$$

Let  $\text{trnc} : \mathbb{R} \rightarrow [-1, 1]$  be the truncation function which on input  $\alpha > 1$ , returns 1,  $\alpha < -1$  returns  $-1$  and otherwise returns  $\alpha$ . This naturally defines a function  $\text{trnc} : \mathbb{R}^{2N} \rightarrow [-1, 1]^{2N}$  obtained by truncating each coordinate. We now define a distribution  $\mathcal{D}$  over  $\{-1, 1\}^{2N}$  generated from  $\mathcal{G}$  by truncating the sample and then independently sampling each coordinate as follows.

1. Sample  $z \in \mathcal{G}$ .
2. For each coordinate  $i \in [2N]$  independently, let  $z'_i = 1$  with probability  $\frac{1+\text{trnc}(z_i)}{2}$  and  $-1$  with probability  $\frac{1-\text{trnc}(z_i)}{2}$ .
3. Output  $z'$ .

We refer to the distribution  $\mathcal{D}$  as the *forrelation* distribution. We state Claim 6.3 from [15] which implies that a vector drawn from this distribution has large forrelation on expectation. The proof is omitted.

► **Lemma 2.** *Let  $\mathcal{D}$  be the forrelation distribution as defined previously. Then,*

$$\mathbb{E}_{z \sim \mathcal{D}}[\text{forr}(z)] \geq \frac{\epsilon}{2}.$$

## 2.5 Multilinear Functions on $\mathcal{D}$

Given a function  $f : \{-1, 1\}^n \rightarrow \mathbb{R}$ , there is a unique multilinear polynomial  $\tilde{f} : \mathbb{R}^n \rightarrow \mathbb{R}$  which agrees with  $f$  on  $\{-1, 1\}^n$ . This polynomial is called the multilinear extension of  $f$ . The multilinear extension of any character function  $\chi_S(x)$  is precisely  $\prod_{i \in S} x_i$ . The multilinear extension  $\tilde{f}$  of  $f$  satisfies  $\tilde{f}(x) = \sum_{S \subseteq [n]} \hat{f}(S) \prod_{i \in S} x_i$  for all  $x \in \mathbb{R}^n$ . We sometimes identify  $f$  with its multilinear extension. The main content of this section is that bounded multilinear functions have similar expectations under  $\mathcal{G}$  and under  $\mathcal{D}$ .

▷ **Claim 3.** Let  $F : \mathbb{R}^{2N} \rightarrow \mathbb{R}$  be any multilinear function  $F = \sum_S \hat{F}(S) \chi_S$ . Then,

$$\mathbb{E}_{z' \sim \mathcal{D}}[F(z')] = \mathbb{E}_{z \sim \mathcal{G}}[F(\text{trnc}(z))].$$

The proof of this claim is identical to that of Equation (2) in [15]. The details can be found in the full version of this paper. The following claim states that  $\mathbb{E}_{z \sim \mathcal{G}}[F(\text{trnc}(z))]$  is pretty close to  $\mathbb{E}_{z \sim \mathcal{G}}[F(z)]$  for a bounded multilinear function  $F$ . Its proof is identical to that in [15], so we omit it. The underlying idea is that  $\epsilon$  is small, so the random variable  $z \sim \mathcal{G}$  has an exponentially decaying norm, furthermore, bounded multilinear functions  $F$  on  $\{-1, 1\}^{2N}$  cannot grow faster than exponentially in the norm of the argument.

▷ **Claim 4.** Let  $F(z)$  be any multilinear polynomial mapping  $\{-1, 1\}^{2N}$  to  $[-1, 1]$ . Let  $z_0 \in [-1/2, 1/2]^{2N}$ ,  $p \leq \frac{1}{2}$  and  $N > 1$ . Then,

$$\mathbb{E}_{z \sim \mathcal{G}}[|F(\text{trnc}(z_0 + pz)) - F(z_0 + pz)|] \leq \frac{8}{N^5}.$$

We remark that the bound in [15] is  $\frac{8}{N^2}$ . The improved bound of  $\frac{8}{N^5}$  in Claim 4 follows from our choice of  $\epsilon = \frac{1}{50 \ln N}$ , as opposed to  $\epsilon = \frac{1}{24 \ln N}$  as in [15].

## 2.6 Moments of $\mathcal{G}$

In this section we state some facts about the moments of the forrelation distribution that will be useful later. We use the following notation to refer to the moments of  $\mathcal{G}$ .

$$\hat{\mathcal{G}}(S, T) := \mathbb{E}_{(x, y) \sim \mathcal{G}} \left[ \prod_{i \in S} x_i \prod_{j \in T} y_j \right]$$



The following claim and its proof are analogous to Claim 4.1 in [15].

▷ **Claim 5.** Let  $S, T \subseteq [N]$  and  $i, j \in [N]$ . Let  $k_1 = |S|, k_2 = |T|$ . Then,

1.  $\widehat{\mathcal{G}}(\{i\}, \{j\}) = \epsilon N^{-1/2} (-1)^{\langle i, j \rangle}$ .
2.  $\widehat{\mathcal{G}}(S, T) = 0$  if  $k_1 \neq k_2$ .
3.  $|\widehat{\mathcal{G}}(S, T)| \leq \epsilon^k k! N^{-k/2}$  if  $k = k_1 = k_2$ .
4.  $|\widehat{\mathcal{G}}(S, T)| \leq \epsilon^{|S|}$  for all  $S, T$ .

### 3 The Forrelation Communication Problem

In this section we formally state the main theorems of this paper.

Let  $\epsilon = \frac{1}{50 \ln N}$  be the parameter as before, defining the forrelation distribution. We restate Theorem 3.

► **Theorem 3.** Consider the following distribution. A string  $z \in \{-1, 1\}^{2N}$  is drawn from the forrelation distribution,  $x \sim U_{2N}$  is drawn uniformly and  $y := x \cdot z$ . Alice gets  $x$  and Bob gets  $y$ . Given any deterministic communication protocol  $C : \{-1, 1\}^{2N} \times \{-1, 1\}^{2N} \rightarrow \{-1, 1\}$  of cost  $c \geq 1$ , its expectation when the inputs are drawn from this distribution is close to when the inputs are drawn from the uniform distribution. That is,

$$\left| \mathbb{E}_{\substack{x \sim U_{2N} \\ z \sim \mathcal{D}}} [C(x, x \cdot z)] - \mathbb{E}_{x, y \sim U_{2N}} [C(x, y)] \right| \leq O\left(\frac{c^2}{N^{1/2}}\right).$$

In other words, no deterministic protocol of cost  $o(N^{1/4})$  has considerable advantage in distinguishing the above distribution from the uniform distribution.

► **Definition 6 (The Forrelation Problem).** Alice is given  $x \in \{-1, 1\}^{2N}$  and Bob is given  $y \in \{-1, 1\}^{2N}$ . Their goal is to compute the partial boolean function  $F$  defined as follows.

$$F(x, y) = \begin{cases} -1 & \text{if } \text{forr}(x \cdot y) \geq \epsilon/4 \\ 1 & \text{if } \text{forr}(x \cdot y) \leq \epsilon/8. \end{cases}$$

We restate Theorem 1 and Theorem 2.

► **Theorem 1.** The forrelation problem can be solved in the quantum simultaneous with entanglement model with  $O(\log^3 N)$  bits of communication, when Alice and Bob are given access to  $O(\log^3 N)$  bits of shared entanglement. Moreover, the protocol is efficient, as it can be implemented by a  $O(\log^3 N)$  size quantum circuit with oracle access to inputs.

The upper bound on the quantum communication complexity of the forrelation problem follows from the fact that the XOR of the inputs can be computed by a simultaneous-message quantum protocol, when the players share entanglement, and the fact that  $\text{forr}(\circ)$  can be estimated by a small size quantum circuit [1, 2, 15]. The proof of Theorem 1 can be found in the full version of this paper.

► **Theorem 2.** The randomized bounded-error interactive communication cost of the forrelation problem is  $\Omega(N^{\frac{1}{4}})$ .

The lower bound on the randomized communication complexity of the forrelation problem follows from Theorem 3 and Lemma 2. The proof of Theorem 2 can be found in the full version of this paper. We now describe the proof of Theorem 3.

#### 4 Proof of Theorem 3 : Distributional Lower Bound

Let  $C : \{-1, 1\}^{2N} \times \{-1, 1\}^{2N} \rightarrow \{-1, 1\}$  be any deterministic protocol of cost at most  $c$ . Let  $D : \{-1, 1\}^{2N} \times \{-1, 1\}^{2N} \rightarrow \{-1, 1\}$  be defined as follows. For  $x, z \in \{-1, 1\}^{2N}$ ,

$$D(x, z) := C(x, x \cdot z).$$

We will also use  $D(x, z)$  to refer to its multilinear extension. Note that our goal is to show that the function  $\mathbb{E}_{x \sim U_{2N}} [D(x, z)]$  of  $z$  is fooled by  $\mathcal{D}$ . Towards this, we will prove that it is fooled by  $p\mathcal{G}$  for small  $p$ . This approach was first used in [6] and is analogous to Claim 7.2 in [15].

► **Lemma 7.** *Let  $p \leq \frac{1}{2N}$  and let  $C(x, y)$  be any deterministic protocol of cost  $c \geq 1$  for the forrelation problem. As before, let  $D(x, z) : \mathbb{R}^{2N} \times \mathbb{R}^{2N} \rightarrow \mathbb{R}$  refer to the multilinear extension of  $C(x, x \cdot z)$ . Let  $P \in [-p, p]^{2N}$ . Then,*

$$\left| \mathbb{E}_{\substack{z \sim P \cdot \mathcal{G} \\ x \sim U_{2N}}} [D(x, z)] - \mathbb{E}_{z, x \sim U_{2N}} [D(x, z)] \right| \leq \frac{120\epsilon c^2 p^2}{\sqrt{N}} + p^4 N^3.$$

**Proof of Lemma 7.** We begin by observing some properties of the distribution  $P \cdot \mathcal{G}$ . The sample  $z \sim P \cdot \mathcal{G}$  is obtained by scaling the  $i$ -th coordinate of  $z' \sim \mathcal{G}$  by  $P_i$  for each  $i \in [2N]$ . This implies that for all  $S \subseteq [2N]$ ,

$$\mathbb{E}_{z \sim P \cdot \mathcal{G}} [\chi_S(z)] = \left( \prod_{i \in S} P_i \right) \mathbb{E}_{z \sim \mathcal{G}} [\chi_S(z)]. \quad (1)$$

Part (2.) of Claim 5 implies that the odd moments of  $\mathcal{G}$  are zero. Equation (1) implies that this is also true for  $P \cdot \mathcal{G}$ . That is, for all  $S \subseteq [2N]$ ,

$$|S| \text{ is odd} \implies \mathbb{E}_{z \sim P \cdot \mathcal{G}} [\chi_S(z)] = 0. \quad (2)$$

Part (3.) of Claim 5 implies that for  $S \subseteq [2N]$ ,  $|S| = 2k$ , the  $S$ -th moment  $\mathbb{E}_{z \sim \mathcal{G}} \chi_S(z)$  is at most  $\epsilon^k k! N^{-k/2}$  in magnitude. Along with equation (1), this implies that for  $k \in \mathbb{N}$ ,

$$|S| = 2k \implies \left| \mathbb{E}_{z \sim P \cdot \mathcal{G}} [\chi_S(z)] \right| \leq \left( \prod_{i \in S} P_i \right) \epsilon^k k! N^{-k/2} \leq p^{2k} \epsilon^k k! N^{-k/2}. \quad (3)$$

We now proceed with the proof of the lemma. Let

$$\Delta := \left| \mathbb{E}_{\substack{z \sim P \cdot \mathcal{G} \\ x \sim U_{2N}}} [D(x, z)] - \mathbb{E}_{z, x \sim U_{2N}} [D(x, z)] \right|.$$

Note that this is the quantity we wish to bound in the lemma. For ease of notation, let  $H : \{-1, 1\}^{2N} \rightarrow [-1, 1]$  be defined at every point  $z \in \{-1, 1\}^{2N}$  by

$$H(z) := \mathbb{E}_{x \sim U_{2N}} [D(x, z)].$$

We identify  $H(z)$  with its multilinear extension. Note that by uniqueness of multilinear extensions, the above equality holds even for  $z \in \mathbb{R}^{2N}$ . This implies that

$$\mathbb{E}_{\substack{z \sim P \cdot \mathcal{G} \\ x \sim U_{2N}}} [D(x, z)] = \mathbb{E}_{z \sim P \cdot \mathcal{G}} [H(z)] \quad \text{and} \quad \mathbb{E}_{z, x \sim U_{2N}} [D(x, z)] = \mathbb{E}_{z \sim U_{2N}} [H(z)].$$

This, along with the definition of  $\Delta$  implies that

$$\Delta = \left| \mathbb{E}_{z \sim P \cdot \mathcal{G}}[H(z)] - \mathbb{E}_{z \sim U_{2N}}[H(z)] \right|.$$

Note that  $H(z) = \sum_S \hat{H}(S) \chi_S(z)$  for all  $z \in \mathbb{R}^{2N}$ . This implies that for all distributions  $\mathcal{Z}$  on  $\mathbb{R}^{2N}$ , we have  $\mathbb{E}_{z \sim \mathcal{Z}}[H(z)] = \sum_S \hat{H}(S) \mathbb{E}_{z \sim \mathcal{Z}}[\chi_S(z)]$ . This implies that

$$\Delta = \left| \sum_{S \subseteq [2N]} \hat{H}(S) \left( \mathbb{E}_{z \sim P \cdot \mathcal{G}}[\chi_S(z)] - \mathbb{E}_{z \sim U_{2N}}[\chi_S(z)] \right) \right|.$$

For any probability distribution, the moment corresponding to the empty set is 1 by definition. For all non empty sets  $S$ , we have  $\mathbb{E}_{z \sim U_{2N}}[\chi_S(z)] = 0$ . Using this fact in the above equality, along with the triangle inequality, we have

$$\Delta = \left| \sum_{\emptyset \neq S \subseteq [2N]} \hat{H}(S) \mathbb{E}_{z \sim P \cdot \mathcal{G}}[\chi_S(z)] \right| \leq \sum_{\emptyset \neq S \subseteq [2N]} \left| \hat{H}(S) \right| \left| \mathbb{E}_{z \sim P \cdot \mathcal{G}}[\chi_S(z)] \right|.$$

We use the bounds from (2) and (3) on the moments of  $P \cdot \mathcal{G}$  to derive the following.

$$\begin{aligned} \Delta &\leq \sum_{\substack{|S|=2k \\ k \geq 1}} \left| \hat{H}(S) \right| p^{2k} \epsilon^k k! N^{-k/2} \\ &= \sum_{k \geq 1} L_{2k}(H) p^{2k} \epsilon^k k! N^{-k/2} \end{aligned}$$

We upper bound  $L_{2k}(H)$  by  $\binom{2N}{2k}$  when  $k \geq 2$ . This implies that

$$\begin{aligned} \Delta &\leq L_2(H) \frac{\epsilon p^2}{\sqrt{N}} + \sum_{k \geq 2} \binom{2N}{2k} p^{2k} \epsilon^k k! N^{-k/2} \\ &\leq L_2(H) \frac{\epsilon p^2}{\sqrt{N}} + \sum_{k \geq 2} \frac{2^{2k} N^{2k}}{(2k)!} p^{2k} \epsilon^k k! N^{-k/2} \\ &\leq L_2(H) \frac{\epsilon p^2}{\sqrt{N}} + \sum_{k \geq 2} N^{3k/2} p^{2k} 4^k \epsilon^k. \end{aligned}$$

In the summation  $\sum_{k \geq 2} N^{3k/2} p^{2k} 4^k \epsilon^k$ , we see that every successive term is smaller than the previous by a factor of at least  $1/4$ . This is because the assumption  $p \leq \frac{1}{2N}$  implies that  $p^2 N^{3/2} \leq p^2 N^2 \leq \frac{1}{4}$  and because  $4\epsilon \leq 1$ . Thus, we can bound this summation by twice the first term, which is  $16p^4 N^3 \epsilon^2$ . This implies that

$$\Delta \leq L_2(H) \frac{\epsilon p^2}{\sqrt{N}} + 32p^4 N^3 \epsilon^2.$$

Since  $\epsilon = \frac{1}{50 \ln N} \leq \frac{1}{32}$ , we may bound  $32p^4 N^3 \epsilon^2$  by  $p^4 N^3$ . This implies that

$$\Delta \leq L_2(H) \frac{\epsilon p^2}{\sqrt{N}} + p^4 N^3.$$

We restate Claim 1 which provides a bound on  $L_2(H)$ .

▷ **Claim 1.** Let  $C(x, y) : \{-1, 1\}^{2N} \times \{-1, 1\}^{2N} \rightarrow \{-1, 1\}$  be any deterministic protocol of cost  $c \geq 1$ , let  $D(x, z) : \mathbb{R}^{2N} \times \mathbb{R}^{2N} \rightarrow \mathbb{R}$  refer to the unique multilinear extension of  $C(x, x \cdot z)$  and  $H : \mathbb{R}^{2N} \rightarrow \mathbb{R}$  be defined by  $H(z) = \mathbb{E}_{x \sim U_{2N}} D(x, z)$ . Then,

$$L_2(H) \triangleq \sum_{|S|=2} |\widehat{H}(S)| \leq 120c^2.$$

This claim along with the preceding inequality implies that

$$\Delta \leq 120c^2 \frac{\epsilon p^2}{\sqrt{N}} + p^4 N^3.$$

This completes the proof of Lemma 7. ◀

**Proof of Claim 1.** In order to bound the level-2 Fourier mass of  $H$ , we will use the following lemma. Its statement and proof appear as “Level- $k$  Inequalities” on Page 259 of “Analysis of Boolean Functions” [12].

► **Lemma 8 (Level- $k$  Inequalities).** *Let  $F : \{-1, 1\}^n \rightarrow \{0, 1\}$  have mean  $\mathbb{E}[F] = \alpha$  and let  $k \in \mathbb{N}$  be at most  $2 \ln(1/\alpha)$ . Then,*

$$\sum_{|S|=k} \left( \widehat{F}(S) \right)^2 \leq \alpha^2 \left( \frac{2e}{k} \ln(1/\alpha) \right)^k.$$

We now show the desired bound on  $L_2(H)$ . Since  $C$  is a deterministic protocol of cost at most  $c$ , it induces a partition of the input space  $\{-1, 1\}^{2N} \times \{-1, 1\}^{2N}$  into at most  $2^c$  rectangles. Let  $\mathcal{P}$  be this partition and let  $A \times B$  index rectangles in  $\mathcal{P}$ , where  $A$  (respectively  $B$ ) is the set of Alice’s (respectively Bob’s) inputs compatible with the rectangle. Let  $C(A \times B) \in \{-1, 1\}$  be the output of the protocol on inputs from a rectangle  $A \times B \in \mathcal{P}$ . For all  $x, y \in \{-1, 1\}^{2N}$ , we have

$$C(x, y) = \sum_{A \times B \in \mathcal{P}} C(A \times B) \mathbb{1}_A(x) \mathbb{1}_B(y).$$

By definition,  $D(x, z) = C(x, x \cdot z)$ . This implies that

$$D(x, z) = \sum_{A \times B \in \mathcal{P}} C(A \times B) \mathbb{1}_A(x) \mathbb{1}_B(x \cdot z).$$

Taking an expectation over  $x \sim U_{2N}$  of the above identity implies that

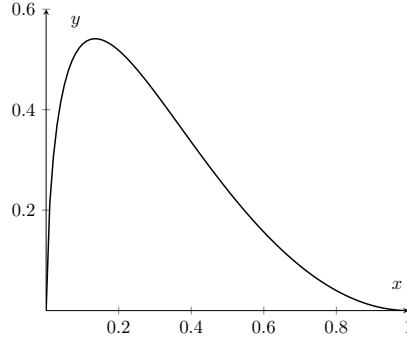
$$H(z) \triangleq \mathbb{E}_{x \sim U_{2N}} [D(x, z)] = \sum_{A \times B \in \mathcal{P}} C(A \times B) (\mathbb{1}_A * \mathbb{1}_B)(z).$$

This implies that for any  $S \subseteq [n]$ , we have

$$\widehat{H}(S) = \sum_{A \times B \in \mathcal{P}} C(A \times B) \widehat{\mathbb{1}_A * \mathbb{1}_B}(S) = \sum_{A \times B \in \mathcal{P}} C(A \times B) \widehat{\mathbb{1}_A}(S) \widehat{\mathbb{1}_B}(S).$$

We thus obtain

$$\begin{aligned} L_2(H) &= \sum_{|S|=2} \left| \widehat{H}(S) \right| \\ &= \sum_{|S|=2} \left| \sum_{A \times B \in \mathcal{P}} C(A \times B) \widehat{\mathbb{1}_A}(S) \widehat{\mathbb{1}_B}(S) \right| \\ &\leq \sum_{A \times B \in \mathcal{P}} \sum_{|S|=2} |\widehat{\mathbb{1}_A}(S)| |\widehat{\mathbb{1}_B}(S)|. \end{aligned}$$



■ **Figure 1** Plot of the function  $y = x \left(\ln \frac{1}{x}\right)^2$ .

We apply Cauchy Schwarz to the term  $\sum_{|S|=2} |\widehat{\mathbb{1}_A}(S)| |\widehat{\mathbb{1}_B}(S)|$  to obtain

$$L_2(H) \leq \sum_{A \times B \in \mathcal{P}} \left( \sum_{|S|=2} \widehat{\mathbb{1}_A}(S)^2 \right)^{1/2} \left( \sum_{|S|=2} \widehat{\mathbb{1}_B}(S)^2 \right)^{1/2}.$$

For ease of notation, let  $\mu(A) = \frac{|A|}{2^{2N}}$  denote the measure of a set  $A \subseteq \{-1, 1\}^{2N}$  under  $U_{2N}$ . We first ensure that for each rectangle  $A \times B \in \mathcal{P}$ , we have  $\mu(A) \leq \frac{1}{e}$  and  $\mu(B) \leq \frac{1}{e}$ . We may do this by adding 2 extra bits of communication for each player. For  $k = 2$ , we have  $k = 2 \ln(e) \leq 2 \ln \frac{1}{\mu(A)}$  and  $k \leq 2 \ln \frac{1}{\mu(B)}$ . We apply Lemma 8 on the indicator functions  $\mathbb{1}_A$  and  $\mathbb{1}_B$  for  $k = 2$  to obtain

$$\sum_{|S|=2} \left( \widehat{\mathbb{1}_A}(S) \right)^2 \leq \mu(A)^2 \left( e \ln(1/\mu(A)) \right)^2 \quad \text{and} \quad \sum_{|S|=2} \left( \widehat{\mathbb{1}_B}(S) \right)^2 \leq \mu(B)^2 \left( e \ln(1/\mu(B)) \right)^2.$$

Substituting this in the bound for  $L_2(H)$ , we have

$$L_2(H) \leq e^2 \sum_{A \times B \in \mathcal{P}} \mu(A) \mu(B) \ln \frac{1}{\mu(A)} \ln \frac{1}{\mu(B)}.$$

Let  $\Delta := e^2 \sum_{A \times B \in \mathcal{P}} \mu(A) \mu(B) \ln \frac{1}{\mu(A)} \ln \frac{1}{\mu(B)}$  be the expression in the R.H.S. of the above.

Note that it suffices to upper bound  $\Delta$ . Consider the case when  $\mathcal{P}$  consists of  $2^c$  rectangles  $A \times B$ , each of which satisfies  $\mu(A) = \mu(B) = \frac{1}{2^{c/2}}$ . In this case,  $\Delta$  evaluates to  $e^2 \sum_{A \times B \in \mathcal{P}} \frac{1}{2^c} \left( \frac{c \ln 2}{2} \right)^2 = O(c^2)$ . This proves the lemma in this special case. A similar bound holds for the general case and the proof follows from a concavity argument that we describe now.

Since  $\mu(A), \mu(B) \leq 1$ , we have the following inequality.

$$\begin{aligned} \Delta &\triangleq e^2 \sum_{A \times B \in \mathcal{P}} \mu(A) \mu(B) \ln \frac{1}{\mu(A)} \ln \frac{1}{\mu(B)} \\ &\leq e^2 \sum_{A \times B \in \mathcal{P}} \mu(A) \mu(B) \ln \frac{1}{\mu(A) \mu(B)} \ln \frac{1}{\mu(A) \mu(B)} \\ &= e^2 \sum_{A \times B \in \mathcal{P}} \mu(A \times B) \left( \ln \frac{1}{\mu(A \times B)} \right)^2. \end{aligned}$$

Let  $f : [0, \infty) \rightarrow \mathbb{R}$  be defined by  $f(p) := p \ln(1/p)^2$ . A small calculation shows that  $f$  is a concave function in the interval  $[0, 0.3]$  (see Figure 7). Let  $\alpha_i \in [0, 0.3]$  for  $i \in [k]$ . Jensen's inequality applied to  $f$  states that for  $i \sim [k]$  drawn uniformly at random, we have  $\mathbb{E}_i[f(\alpha_i)] \leq f(\mathbb{E}_i[\alpha_i])$ . This implies that

$$\sum_{i=1}^k \alpha_i \ln(1/\alpha_i)^2 \leq \left( \sum_{i=1}^k \alpha_i \right) \ln \left( \frac{k}{\sum_{i=1}^k \alpha_i} \right)^2.$$

We apply this inequality to the terms in  $\Delta$  by substituting  $\alpha_i$  with  $\mu(A \times B)$ . We may do this since the assumption that  $\mu(A), \mu(B) \leq \frac{1}{e}$  implies that  $\mu(A \times B) \leq \frac{1}{e^2} \leq 0.3$ . This implies that

$$\Delta \leq e^2 \left( \sum_{A \times B \in \mathcal{P}} \mu(A \times B) \right) \ln \left( \frac{2^{c+4}}{\sum_{A \times B \in \mathcal{P}} \mu(A \times B)} \right)^2$$

Since  $\sum_{A \times B \in \mathcal{P}} \mu(A \times B) = 1$ , we have

$$\Delta \leq e^2 (c+4)^2 (\ln 2)^2 \leq 120c^2.$$

This completes the proof of Claim 1.  $\triangleleft$

We now show that an analogue of Lemma 7 holds for restricted protocols, similarly to Claim 7.3 in [15].

► **Lemma 9.** *Let  $p \leq \frac{1}{4N}$  and  $C(x, y)$  be any deterministic protocol of cost  $c \geq 1$  for the forrelation problem. As before, let  $D(x, z) : \mathbb{R}^{2N} \times \mathbb{R}^{2N} \rightarrow \mathbb{R}$  refer to the multilinear extension of  $C(x, x \cdot z)$ . Let  $z_0 \in [-1/2, 1/2]^{2N}$ . Then,*

$$\left| \mathbb{E}_{\substack{z \sim p\mathcal{G} \\ x \sim U_{2N}}} [D(x, z_0 + z)] - \mathbb{E}_{z, x \sim U_{2N}} [D(x, z_0 + z)] \right| \leq \frac{120\epsilon c^2 (2p)^2}{\sqrt{N}} + (2p)^4 N^3.$$

► **Corollary 10.** *Under the same hypothesis as in Lemma 9,*

$$\left| \mathbb{E}_{z \sim p\mathcal{G}} [D(0, z_0 + z)] - D(0, z_0) \right| \leq \frac{120\epsilon c^2 (2p)^2}{\sqrt{N}} + (2p)^4 N^3.$$

**Proof of Corollary 10 from Lemma 9.** Since  $D(x, z)$  is a multilinear polynomial, for all  $z \in \mathbb{R}^{2N}$ , we have  $\mathbb{E}_{x \sim U_{2N}} [D(x, z)] = D(0, z)$ . This implies that for all  $z_0 \in \mathbb{R}^{2N}$ ,

$$\mathbb{E}_{\substack{z \sim p\mathcal{G} \\ x \sim U_{2N}}} [D(x, z_0 + z)] = \mathbb{E}_{z \sim p\mathcal{G}} [D(0, z_0 + z)].$$

For all  $z_0 \in \mathbb{R}^{2N}$ , since  $\mathbb{E}_{z \sim U_{2N}} [D(0, z_0 + z)] = D(0, z_0)$ , we have

$$\mathbb{E}_{z, x \sim U_{2N}} [D(x, z_0 + z)] = D(0, z_0).$$

The proof of Corollary 10 follows from the above two equalities and Lemma 9.  $\blacktriangleleft$

**Proof of Lemma 9.** Similarly to the approach of [6, 15], we will express  $D(x, z_0 + z)$  as the average output of restricted protocols  $(C \circ \rho)(x, x \cdot z)$ , on which we can use Lemma 7 to derive the result. These restricted protocols roughly correspond to Alice and Bob fixing a common subset  $I \subseteq [2N]$  of their inputs in a predetermined way and then running the original protocol. We formalize this now.

A restriction  $\rho$  of  $\mathbb{R}^{2N}$  is an element of  $\{-1, 1, *\}^{2N}$ . It defines an action  $\rho : \mathbb{R}^{2N} \rightarrow \mathbb{R}^{2N}$  in the following natural way. For any  $z \in \mathbb{R}^{2N}$  and  $i \in [2N]$ ,

$$(\rho(z))(i) := \begin{cases} \rho(i) & \text{if } \rho(i) \in \{-1, 1\} \\ z(i) & \text{otherwise.} \end{cases}$$

Let  $\text{sign} : (\mathbb{R} \setminus 0) \rightarrow \{-1, 1\}$  be the function which maps real numbers to their sign. Given  $z_0 \in [-1/2, 1/2]^{2N}$ , let  $R_{z_0}$  be a distribution over restrictions of  $\mathbb{R}^{2N}$  defined as follows. For each  $i \in [2N]$ , independently, set<sup>1</sup>:

$$\rho(i) := \begin{cases} \text{sign}(z_0(i)) & \text{with probability } |z_0(i)| \\ * & \text{with probability } 1 - |z_0(i)|. \end{cases}$$

Let  $P \in \mathbb{R}^{2N}$  be such that  $P_i := \frac{1}{1-|z_0(i)|}$  for every  $i \in [2N]$ . Note that the assumption of  $z_0 \in [-1/2, 1/2]^{2N}$  ensures that  $P$  is a well defined element of  $[1, 2]^{2N}$ . For any  $z \in \mathbb{R}^{2N}$  and  $i \in [2N]$ , the expected value of the  $i$ th coordinate of  $\rho(z)$  when  $\rho \sim R_{z_0}$  can be computed as follows.

$$\mathbb{E}_{\rho \sim R_{z_0}} [(\rho(z))(i)] = |z_0(i)| \text{sign}(z_0(i)) + (1 - |z_0(i)|)z(i) = z_0(i) + \frac{1}{P_i}z(i)$$

This implies that for any fixed  $x, z \in \mathbb{R}^{2N}$  and  $z_0 \in [-1/2, 1/2]^{2N}$ , since  $D$  is a multilinear function, we have

$$\mathbb{E}_{\rho \sim R_{z_0}} [D(x, \rho(z))] = D(x, \mathbb{E}_{\rho \sim R_{z_0}} [\rho(z)]) = D(x, z_0 + P^{-1} \cdot z).$$

Replacing  $z$  with  $P \cdot z$  in the above equality implies that

$$\mathbb{E}_{\rho \sim R_{z_0}} [D(x, \rho(P \cdot z))] = D(x, z_0 + z).$$

This equality allows us to rewrite the L.H.S. of Lemma 9 as follows.

$$\begin{aligned} \Delta &:= \left| \mathbb{E}_{\substack{z \sim p\mathcal{G}, \\ x \sim U_{2N}}} [D(x, z_0 + z)] - \mathbb{E}_{z, x \sim U_{2N}} [D(x, z_0 + z)] \right| \\ &= \left| \mathbb{E}_{\substack{z \sim pP \cdot \mathcal{G}, \\ x \sim U_{2N}}} \mathbb{E}_{\rho \sim R_{z_0}} [D(x, \rho(z))] - \mathbb{E}_{\substack{z \sim P \cdot U_{2N}, \\ x \sim U_{2N}}} \mathbb{E}_{\rho \sim R_{z_0}} [D(x, \rho(z))] \right| \\ &= \left| \mathbb{E}_{\rho \sim R_{z_0}} \left[ \mathbb{E}_{\substack{z \sim pP \cdot \mathcal{G}, \\ x \sim U_{2N}}} [D(x, \rho(z))] - \mathbb{E}_{\substack{z \sim P \cdot U_{2N}, \\ x \sim U_{2N}}} [D(x, \rho(z))] \right] \right|. \end{aligned}$$

For a multilinear polynomial, its expectation over a product distribution depends only on the mean of that distribution. This allows us to replace the expectation of  $D(x, \rho(z))$  over  $z \sim P \cdot U_{2N}$  by an expectation over  $z \sim U_{2N}$ . We thus obtain

$$\Delta = \left| \mathbb{E}_{\rho \sim R_{z_0}} \left[ \mathbb{E}_{\substack{z \sim pP \cdot \mathcal{G}, \\ x \sim U_{2N}}} [D(x, \rho(z))] - \mathbb{E}_{\substack{z \sim U_{2N}, \\ x \sim U_{2N}}} [D(x, \rho(z))] \right] \right| \quad (4)$$

---

<sup>1</sup> If  $z_0(i)$  is zero, then  $\rho(i) = *$  with probability 1.



For any  $\rho \in \{-1, 1, *\}^{2N}$  and  $u \in \{-1, 1\}^{2N}$ , we define a substitution  $\rho^u : \mathbb{R}^{2N} \rightarrow \mathbb{R}^{2N}$  obtained from  $\rho$  and  $u$  as follows. For any  $x \in \mathbb{R}^{2N}$  and  $i \in [2N]$ ,

$$(\rho^u(x))(i) := \begin{cases} u(i) & \text{if } \rho(i) \in \{-1, 1\} \\ x(i) & \text{otherwise.} \end{cases}$$

This is an action on  $\mathbb{R}^{2N}$  which replaces the values of coordinates specified by  $\rho$ , with values from  $u$ . For every fixed  $\rho$ , as we vary over  $x, u \sim U_{2N}$  the distribution of  $\rho^u(x)$  is exactly  $U_{2N}$ . This implies that for all  $z \in \mathbb{R}^{2N}, \rho \in \{-1, 1, *\}^{2N}$ ,

$$\mathbb{E}_{x \sim U_{2N}} [D(x, \rho(z))] = \mathbb{E}_{x, u \sim U_{2N}} [D(\rho^u(x), \rho(z))].$$

Substituting this in equation (4), we have

$$\Delta = \left| \mathbb{E}_{\rho \sim R_{z_0}} \mathbb{E}_{u \sim U_{2N}} \left[ \mathbb{E}_{\substack{z \sim pP \cdot \mathcal{G}, \\ x \sim U_{2N}}} [D(\rho^u(x), \rho(z))] - \mathbb{E}_{z, x \sim U_{2N}} [D(\rho^u(x), \rho(z))] \right] \right|.$$

Applying Triangle Inequality on the above, we have

$$\Delta \leq \mathbb{E}_{\rho \sim R_{z_0}} \mathbb{E}_{u \sim U_{2N}} \left| \mathbb{E}_{\substack{z \sim pP \cdot \mathcal{G}, \\ x \sim U_{2N}}} [D(\rho^u(x), \rho(z))] - \mathbb{E}_{z, x \sim U_{2N}} [D(\rho^u(x), \rho(z))] \right|. \quad (5)$$

Fix any  $\rho \in \{-1, 1, *\}^{2N}$  and  $u \in \{-1, 1\}^{2N}$ . For every  $x, z \in \{-1, 1\}^{2N}$ , we have  $D(x, z) = C(x, x \cdot z)$ , furthermore,  $\rho^u(x), \rho(z) \in \{-1, 1\}^{2N}$ . This implies that for every  $x, z \in \{-1, 1\}^{2N}$ ,

$$D(\rho^u(x), \rho(z)) = C(\rho^u(x), \rho^u(x) \cdot \rho(z)). \quad (6)$$

This prompts us to define a communication protocol  $C \circ \rho^u$  where Alice and Bob first restrict their inputs and then run the original protocol  $C$ . The restriction is that for each coordinate  $i \in [2N]$  with  $\rho_i \in \{-1, 1\}$ , Alice overwrites her input  $x_i$  with  $u_i$  while Bob overwrites his input  $y_i$  with  $\rho_i u_i$ . The main property of this restricted protocol is that for all  $x, z \in \{-1, 1\}^{2N}$ ,

$$(C \circ \rho^u)(x, x \cdot z) = C(\rho^u(x), \rho^u(x) \cdot \rho(z)).$$

This, along with equation (6) implies that  $D(\rho^u(x), \rho(z))$  is the unique multilinear extension of  $(C \circ \rho^u)(x, x \cdot z)$ . The cost of  $C \circ \rho^u$  is at most that of  $C$  since Alice and Bob don't need to communicate to restrict their inputs. We now use Lemma 7 on  $C \circ \rho^u$  to argue that  $pP \cdot \mathcal{G}$  fools  $\mathbb{E}_{x \sim U_{2N}} [D(\rho^u(x), \rho(z))]$ . The conditions of the lemma are satisfied since  $pP \in [-2p, 2p]^{2N}$ ,  $p \leq \frac{1}{4N}$ , and  $C \circ \rho^u$  is a protocol of cost at most  $c$  and whose multilinear extension is  $D(\rho^u(x), \rho(z))$ . The lemma implies that

$$\left| \mathbb{E}_{\substack{z \sim pP \cdot \mathcal{G}, \\ x \sim U_{2N}}} [D(\rho^u(x), \rho(z))] - \mathbb{E}_{\substack{z \sim U_{2N}, \\ x \sim U_{2N}}} [D(\rho^u(x), \rho(z))] \right| \leq \frac{120\epsilon c^2 (2p)^2}{\sqrt{N}} + (2p)^4 N^3.$$

Substituting this in inequality (5) completes the proof of Lemma 9. ◀

**Proof of Theorem 3.** Since  $D(x, z)$  is the multilinear extension of  $C(x, x \cdot z)$  and since  $\mathcal{D}$  and  $U_{2N}$  are distributions over  $\{-1, 1\}^{2N}$ , we have

$$\mathbb{E}_{x \sim U_{2N}, z \sim \mathcal{D}}[C(x, x \cdot z)] = \mathbb{E}_{x \sim U_{2N}, z \sim \mathcal{D}}[D(x, z)] = \mathbb{E}_{z \sim \mathcal{D}}[D(0, z)].$$

When  $x \sim U_{2N}$  and  $y \sim U_{2N}$  are independently sampled, the distribution of  $(x, x \cdot y)$  is  $U_{4N}$ . This implies that

$$\mathbb{E}_{x, y \sim U_{2N}}[C(x, y)] = \mathbb{E}_{x, y \sim U_{2N}}[D(x, x \cdot y)] = D(0, 0).$$

The above two equations allow us to rewrite the quantity in the L.H.S. of Theorem 3 as follows.

$$\Delta := \left| \mathbb{E}_{\substack{x \sim U_{2N} \\ z \sim \mathcal{D}}} [C(x, x \cdot z)] - \mathbb{E}_{x, y \sim U_{2N}} [C(x, y)] \right| = \left| \mathbb{E}_{z \sim \mathcal{D}} [D(0, z)] - D(0, 0) \right|$$

Claim 3 applied on the multilinear polynomial  $D$  implies that  $\mathbb{E}_{z \sim \mathcal{D}} [D(0, z)] = \mathbb{E}_{z \sim \mathcal{G}} [D(0, \text{trnc}(z))]$ . Substituting this in the above equality implies that

$$\Delta = \left| \mathbb{E}_{z \sim \mathcal{G}} [D(0, \text{trnc}(z))] - D(0, 0) \right|.$$

Let  $t = 16N^4$ ,  $p = \frac{1}{\sqrt{t}} = \frac{1}{4N^2}$ . Let  $z^{(1)}, \dots, z^{(t)} \sim \mathcal{G}$  be independent samples and let  $Z$  refer to this collection of random variables. For  $i \in [t]$ , define  $z^{\leq(i)} := p(z^{(1)} + \dots + z^{(i)})$ . By convention,  $z^{\leq(0)} := 0$ . Note that for  $i \in [t]$ ,  $z^{\leq(i)}$  has a Gaussian distribution with mean 0 and covariance matrix as  $p^2 i$  times that of  $\mathcal{G}$ . Thus,  $z^{\leq(t)}$  is sampled according to  $\mathcal{G}$ . Substituting this in the previous equality implies that

$$\Delta = \left| \mathbb{E}_Z [D(0, \text{trnc}(z^{\leq t}))] - D(0, 0) \right|.$$

To bound the above quantity, for each  $0 \leq i \leq t-1$ , we show a bound on

$$\Delta_i := \left| \mathbb{E}_Z [D(0, \text{trnc}(z^{\leq(i+1)}))] - \mathbb{E}_Z [D(0, \text{trnc}(z^{\leq(i)}))] \right|.$$

Since  $z^{\leq(0)} = 0$ , the triangle inequality implies that  $\Delta \leq \sum_{i=0}^{t-1} \Delta_i$ .

Fix any  $i \in \{0, \dots, t-1\}$ . We now bound  $\Delta_i$ . Let  $E_i$  be the event that  $z^{\leq(i)} \notin [-1/2, 1/2]^{2N}$ . We first observe that  $E_i$  is a low probability event. Since each  $z^{\leq(i)}(j)$  is distributed as  $\mathcal{N}(0, p^2 i \epsilon)$ , where  $p^2 i \leq 1$  and  $\epsilon = 1/(50 \ln N)$ , we have

$$\mathbb{P}[z^{\leq(i)}(j) \notin [-1/2, 1/2]] \leq \mathbb{P}[|\mathcal{N}(0, \epsilon)| \geq 1/2] \leq \exp(-1/8\epsilon) \leq \exp(-6 \ln N) = \frac{1}{N^6}$$

Applying a Union bound over coordinates  $j \in [2N]$ , we have for each  $0 \leq i \leq t$ ,

$$\mathbb{P}[E_i] = \mathbb{P}[z^{\leq(i)} \notin [-1/2, 1/2]^{2N}] \leq 2N \frac{1}{N^6} \leq \frac{2}{N^5}. \quad (7)$$

When  $E_i$  does not occur, we have  $\text{trnc}(z^{\leq(i)}) = z^{\leq(i)} \in [-1/2, 1/2]^{2N}$ . For every fixed value of  $z^{\leq(i)}$  in this range, we apply Corollary 10 with parameters  $p = \frac{1}{4N^2}$ ,  $z_0 = z^{\leq(i)}$  and  $z = z^{\leq(i+1)} - z^{\leq(i)} = pz^{(i+1)}$ . Note that the conditions in the hypothesis are satisfied since  $z_0 \in [-1/2, 1/2]^{2N}$ ,  $p \leq 1/(4N)$  and the random variable  $pz^{(i+1)}$  is distributed as  $p\mathcal{G}$ . The corollary implies that for every  $z^{\leq(i)} \in [-1/2, 1/2]^{2N}$ ,

$$\left| \mathbb{E}_Z [D(0, z^{\leq(i+1)}) \mid z^{\leq(i)}] - \mathbb{E}_Z [D(0, z^{\leq(i)}) \mid z^{\leq(i)}] \right| \leq \frac{120\epsilon c^2 (2p)^2}{N^{1/2}} + (2p)^4 N^3.$$

Since  $\neg E_i$  implies that  $z^{\leq(i)} \in [-1/2, 1/2]^{2N}$ , we have

$$\left| \mathbb{E}_Z \left[ D(0, z^{\leq(i+1)}) \mid \neg E_i \right] - \mathbb{E}_Z \left[ D(0, z^{\leq(i)}) \mid \neg E_i \right] \right| \leq \frac{120\epsilon c^2 (2p)^2}{N^{1/2}} + (2p)^4 N^3.$$

We apply Claim 4 on the multilinear polynomial  $D(0, z) : [-1, 1]^{2N} \rightarrow [-1, 1]$  with the parameters  $p = \frac{1}{4N^2}$ ,  $z_0 = z^{\leq(i)}$  and  $z = z^{\leq(i+1)}$ . Note that the conditions are satisfied since  $z_0 \in [1/2, 1/2]^{2N}$  and  $p \leq \frac{1}{2}$ . The claim implies that

$$\left| \mathbb{E}_Z \left[ D(0, z^{\leq(i+1)}) \mid \neg E_i \right] - \mathbb{E}_Z \left[ D(0, \text{trnc}(z^{\leq(i+1)})) \mid \neg E_i \right] \right| \leq \frac{8}{N^5}.$$

The previous two inequalities, along with the triangle inequality, imply that

$$\left| \mathbb{E}_Z \left[ D(0, \text{trnc}(z^{\leq(i+1)})) \mid \neg E_i \right] - \mathbb{E}_Z \left[ D(0, z^{\leq(i)}) \mid \neg E_i \right] \right| \leq \frac{120\epsilon c^2 (2p)^2}{N^{1/2}} + (2p)^4 N^3 + \frac{8}{N^5}. \quad (8)$$

Note that for every possible values of  $z^{\leq(i+1)}$  and  $z^{\leq(i)}$ , the difference  $D(0, \text{trnc}(z^{\leq(i+1)})) - D(0, \text{trnc}(z^{\leq(i)}))$  is bounded in magnitude by 2, since  $D(0, \text{trnc}(z))$  maps  $\mathbb{R}^{2N}$  to  $[-1, 1]$ . This implies that

$$\left| \mathbb{E}_Z \left[ D(0, \text{trnc}(z^{\leq(i+1)})) \mid E_i \right] - \mathbb{E}_Z \left[ D(0, \text{trnc}(z^{\leq(i)})) \mid E_i \right] \right| \leq 2.$$

Thus, we have

$$\begin{aligned} \Delta_i &\leq \mathbb{P}[\neg E_i] \cdot \left| \mathbb{E}_Z[D(0, \text{trnc}(z^{\leq(i+1)})) \mid \neg E_i] - \mathbb{E}_Z[D(0, \text{trnc}(z^{\leq(i)})) \mid \neg E_i] \right| \\ &\quad + \mathbb{P}[E_i] \cdot \left| \mathbb{E}_Z[D(0, \text{trnc}(z^{\leq(i+1)})) \mid E_i] - \mathbb{E}_Z[D(0, \text{trnc}(z^{\leq(i)})) \mid E_i] \right| \\ &\leq \left| \mathbb{E}_Z[D(0, \text{trnc}(z^{\leq(i+1)})) \mid \neg E_i] - \mathbb{E}_Z[D(0, \text{trnc}(z^{\leq(i)})) \mid \neg E_i] \right| + 2\mathbb{P}[E_i] \\ &= \left| \mathbb{E}_Z[D(0, \text{trnc}(z^{\leq(i+1)})) \mid \neg E_i] - \mathbb{E}_Z[D(0, z^{\leq(i)}) \mid \neg E_i] \right| + 2\mathbb{P}[E_i] \\ &\leq \frac{120\epsilon c^2 (2p)^2}{N^{1/2}} + (2p)^4 N^3 + \frac{8}{N^5} + \frac{4}{N^5}. \end{aligned}$$

The equality in the fourth line follows from the fact that whenever  $E_i$  does not occur,  $\text{trnc}(z^{\leq(i)}) = z^{\leq(i)}$  by definition. The last inequality follows from inequalities (7) and (8). Along with the fact that  $t = \frac{1}{p^2} = 16N^4$ , and  $\epsilon \leq 1$ , this implies that

$$\begin{aligned} \Delta &\leq \sum_{i=0}^{t-1} \Delta_i \leq t \left( \frac{120\epsilon c^2 (2p)^2}{N^{1/2}} + (2p)^4 N^3 + \frac{12}{N^5} \right) \leq \frac{480\epsilon c^2}{N^{1/2}} + 16p^2 N^3 + \frac{192}{N} \\ &= O\left(\frac{c^2}{N^{1/2}} + \frac{1}{N}\right) = O\left(\frac{c^2}{N^{1/2}}\right). \end{aligned}$$

The last inequality follows from the assumption that  $c \geq 1$ . This completes the proof of Theorem 3.  $\blacktriangleleft$

## References

- 1 Scott Aaronson. BQP and the polynomial hierarchy. In *STOC 2010*. ACM, 2010. doi:10.1145/1806689.1806711.
- 2 Scott Aaronson and Andris Ambainis. Forrelation: A problem that optimally separates quantum from classical computing. In *STOC 2015*. ACM, 2015. doi:10.1145/2746539.2746547.

- 3 Ziv Bar-Yossef, T. S. Jayram, and Iordanis Kerenidis. Exponential separation of quantum and classical one-way communication complexity. In *STOC 2004*. ACM, 2004. doi:10.1145/1007352.1007379.
- 4 Harry Buhrman, Richard Cleve, and Avi Wigderson. Quantum vs. classical communication and computation. In *STOC 1998*. ACM, 1998. doi:10.1145/276698.276713.
- 5 Arkadev Chattopadhyay, Yuval Filmus, Sajin Korothe, Or Meir, and Toniann Pitassi. Query-to-communication lifting for BPP using inner product. In *ICALP 2019*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.ICALP.2019.35.
- 6 Eshan Chattopadhyay, Pooya Hatami, Kaave Hosseini, and Shachar Lovett. Pseudorandom generators from polarizing random walks. In *CCC 2018*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.CCC.2018.1.
- 7 Eshan Chattopadhyay, Pooya Hatami, Shachar Lovett, and Avishay Tal. Pseudorandom generators from the second fourier level and applications to AC0 with parity gates. In *ITCS 2019*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.ITCS.2019.22.
- 8 Dmitry Gavinsky. Entangled simultaneity versus classical interactivity in communication complexity. In *STOC 2016, Cambridge, MA, USA, June 18-21, 2016*. ACM, 2016. doi:10.1145/2897518.2897545.
- 9 Dmitry Gavinsky, Julia Kempe, Iordanis Kerenidis, Ran Raz, and Ronald de Wolf. Exponential separations for one-way quantum communication complexity, with applications to cryptography. In *STOC 2007*. ACM, 2007. doi:10.1145/1250790.1250866.
- 10 Mika Göös, Toniann Pitassi, and Thomas Watson. Query-to-communication lifting for BPP. In *FOCS 2017*. IEEE Computer Society, 2017. doi:10.1109/FOCS.2017.21.
- 11 Hamed Hatami, Kaave Hosseini, and Shachar Lovett. Structure of protocols for XOR functions. In *FOCS 2016*. IEEE Computer Society, 2016. doi:10.1109/FOCS.2016.38.
- 12 Ryan O'Donnell. *Analysis of Boolean Functions*. Cambridge University Press 2014, 2014.
- 13 Ran Raz. Fourier analysis for probabilistic communication complexity. *Comput. Complex.*, 1995. doi:10.1007/BF01206318.
- 14 Ran Raz. Exponential separation of quantum and classical communication complexity. In *STOC 1999*. ACM, 1999. doi:10.1145/301250.301343.
- 15 Ran Raz and Avishay Tal. Oracle separation of BQP and PH. In *STOC 2019*. ACM, 2019. doi:10.1145/3313276.3316315.
- 16 Oded Regev and Bo'az Klartag. Quantum one-way communication can be exponentially stronger than classical communication. In *STOC 2011*. ACM, 2011. doi:10.1145/1993636.1993642.

# Agnostic Learning with Unknown Utilities

**Kush Bhatia**

University of California at Berkeley, CA, USA  
kushbhatia@berkeley.edu

**Peter L. Bartlett**

University of California at Berkeley, CA, USA  
peter@berkeley.edu

**Anca D. Dragan**

University of California at Berkeley, CA, USA  
anca@berkeley.edu

**Jacob Steinhardt**

University of California at Berkeley, CA, USA  
jsteinhardt@berkeley.edu

---

## Abstract

Traditional learning approaches for classification implicitly assume that each mistake has the same cost. In many real-world problems though, the utility of a decision depends on the underlying context  $x$  and decision  $y$ ; for instance, misclassifying a stop sign is worse than misclassifying a road-side postbox. However, directly incorporating these utilities into the learning objective is often infeasible since these can be quite complex and difficult for humans to specify.

We formally study this as *agnostic learning with unknown utilities*: given a dataset  $S = \{x_1, \dots, x_n\}$  where each data point  $x_i \sim \mathcal{D}_x$  from some unknown distribution  $\mathcal{D}_x$ , the objective of the learner is to output a function  $f$  in some class of decision functions  $\mathcal{F}$  with small excess risk. This risk measures the performance of the output predictor  $f$  with respect to the best predictor in the class  $\mathcal{F}$  on the *unknown* underlying utility  $u^* : \mathcal{X} \times \mathcal{Y} \mapsto [0, 1]$ . This utility  $u^*$  is not assumed to have any specific structure and is allowed to be any bounded function. This raises an interesting question whether learning is even possible in our setup, given that obtaining a generalizable estimate of utility  $u^*$  might not be possible from finitely many samples. Surprisingly, we show that estimating the utilities of only the sampled points  $S$  suffices to learn a decision function which generalizes well.

With this insight, we study mechanisms for eliciting information from human experts which allow a learner to estimate the utilities  $u^*$  on the set  $S$ . While humans find it difficult to directly provide utility values reliably, it is often easier for them to provide comparison feedback based on these utilities. We show that, unlike in the realizable setup, the vanilla comparison queries where humans compare a pair of decisions for a single input  $x$  are insufficient. We introduce a family of elicitation mechanisms by generalizing comparisons, called the  $k$ -comparison oracle, which enables the learner to ask for comparisons across  $k$  different inputs  $x$  at once. We show that the excess risk in our agnostic learning framework decreases at a rate of  $O\left(\frac{1}{k}\right)$  with such queries. This result brings out an interesting accuracy-elicitation trade-off – as the order  $k$  of the oracle increases, the comparative queries become harder to elicit from humans but allow for more accurate learning.

**2012 ACM Subject Classification** Computing methodologies → Machine learning; Computing methodologies → Active learning settings

**Keywords and phrases** agnostic learning, learning by comparisons, utility estimation, active learning

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.55



© Kush Bhatia, Peter L. Bartlett, Anca D. Dragan, and Jacob Steinhardt;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 55; pp. 55:1–55:20



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

## 1 Introduction

Our focus is on learning predictive models for decision-making tasks. Current paradigms for classification tasks use datasets consisting of scenarios<sup>1</sup>  $x$  along with the decisions  $y$  taken by human experts to learn a decision function<sup>2</sup>  $f : \mathcal{X} \mapsto \mathcal{Y}$ . For instance, in economics such decisions correspond to whether buyers bought an item at a suggested price [2, 9], in robotics such feedback comprises expert demonstrations in imitation learning [1, 4], and in machine learning literature such supervision consists of labels selected by human annotators [10, 13].

When we optimize models to predict correctly on these datasets, we often implicitly assume that all mistakes are equally costly, and that each scenario  $x$  in the data is just as important. In reality though, this is rarely the case. For instance, the standard 0 – 1 loss for classification tasks assigns a unit of loss for each mistake, but misclassifying a stop sign is significantly more dangerous than misclassifying a road-side postbox. In Figure 1, we expand on this insight and illustrate how learning from such revealed decisions can often lead to suboptimal decision functions.

What is missing from this classical framework is that for most decision-making tasks there exists an underlying function  $u^* : \mathcal{X} \times \mathcal{Y} \mapsto [0, 1]$  which evaluates the utility of a decision  $y$  depending on the surrounding context  $x$ . Depending on the decision task, such utility functions can encode buyer preferences in economics, rewards for robotic skills, or misprediction costs for classification. However, these utility functions are a priori unknown to the learner since the dataset consists only of context-decision pairs  $(x, y)$ . Furthermore, asking human experts to write down these complex utility functions can be quite challenging and prone to serious errors [3].

One commonly studied approach, referred to as learning from revealed preferences in economics [9, 6] and inverse reinforcement learning (IRL) in the machine learning literature [19, 26], assumes that the utility function  $u^*$  belongs to some pre-specified class and uses the fact that decision  $y$  was the optimal decision for scenario  $x$  to learn estimates of these utilities. This setup is called the well-specified or realizable setup. However, this posited utility class can be misspecified in that the underlying utility  $u^*$  might not belong to this class. The correctness of such learning approaches crucially relies on the well specified assumption and offers no guarantees on how their performance degrades in the presence of class misspecifications.

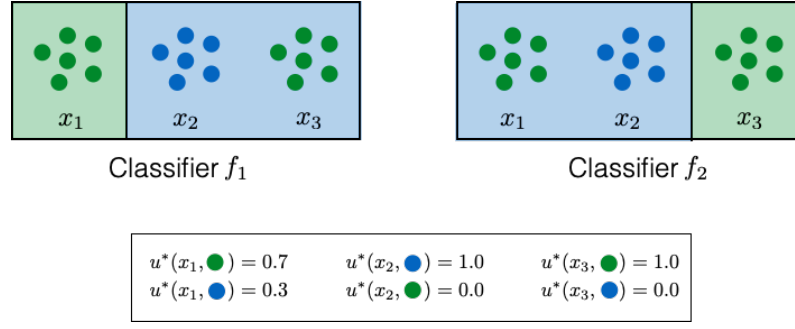
We overcome this uncertainty in specifying the utility function  $u^*$  by proposing an *agnostic learning* framework which places no assumptions on the class of utility functions. Instead, we consider decision functions belonging to some class  $\mathcal{F} = \{f \mid f : \mathcal{X} \mapsto \mathcal{Y}\}$  and study the objective of obtaining the “best” decision rule in  $\mathcal{F}$  with respect to the unknown utility  $u^*$ . Formally, given the decision class  $\mathcal{F}$  and samples from a distribution  $\mathcal{D}_x$  over the feature space  $\mathcal{X}$ , the objective of the learner is to output a model  $\hat{f} \in \mathcal{F}$  with small excess risk or regret

$$\text{err}(\hat{f}, \mathcal{F}) := \sup_{f \in \mathcal{F}} \mathbb{E}_{x \sim \mathcal{D}_x} [u^*(x, f(x))] - \mathbb{E}_{x \sim \mathcal{D}_x} [u^*(x, \hat{f}(x))] . \quad (1)$$

Our proposed notion of excess risk measures the performance of an estimator  $\hat{f}$  by comparing its decisions with those of the best predictive model in the class  $\mathcal{F}$  under the utility  $u^*$ .

<sup>1</sup> We use the term scenario/context/feature for the vector  $x$  interchangeably.

<sup>2</sup> We consider finite decision spaces  $\mathcal{Y}$ .



**Figure 1** Consider a binary decision-task with decisions G(reen) and B(lue). The instance space comprises of three equiprobable clusters of datapoints  $x_1, x_2$  and  $x_3$ , and have associated utilities  $u^*$  for decisions B and G. The colour of the datapoints represents the decision with higher utility. The function class  $\mathcal{F}$  consists of linear predictors. In the traditional learning setups where the dataset consists of pairs  $(x, y)$ , no learner will have enough information to select between  $f_1$  and  $f_2$  since the 0 – 1 error for both is  $1/3$ . In contrast, using a 2-comparison oracle, a learner can ask a query of the form “Which of  $u^*(x_1, \text{G}) + u^*(x_3, \text{B})$  or  $u^*(x_1, \text{B}) + u^*(x_3, \text{G})$  is bigger?”. This allows them to infer that correctly predicting  $x_3$  gives a higher overall utility and output the optimal decision function  $f_2$ .

Contrast this with the classical agnostic learning framework [15] where the evaluation metric for classification measures what proportion of datapoints  $\hat{f}$  predicts correctly

$$\text{err}_{\text{cl}}(\hat{f}, \mathcal{F}) := \sup_{f \in \mathcal{F}} \mathbb{E}_{x \sim \mathcal{D}_x} [\mathbb{I}[f(x) \neq y_x]] - \mathbb{E}_{x \sim \mathcal{D}_x} [\mathbb{I}[\hat{f}(x) \neq y_x]], \quad (2)$$

where  $y_x = \arg\max_{y \in \mathcal{Y}} u^*(x, y)$  represents the expert decision (revealed decision) for scenario  $x$ . Our above framework generalizes the proper agnostic learning framework – we restrict our attention to proper learners which output models  $\hat{f} \in \mathcal{F}$  and the decision class  $\mathcal{F}$  is agnostic towards the unknown underlying utility  $u^*$ . Indeed, our agnostic framework allows for misspecification in the decision class  $\mathcal{F}$  and allows for situations where no predictive model  $f \in \mathcal{F}$  matches the expert predictions  $y_x$  for all instances  $x$ .

As highlighted by Figure 1, such a misspecification in the function class  $\mathcal{F}$  implies that no decision function  $f \in \mathcal{F}$  will be able to perfectly fit these optimal decisions  $y_x$  for all points  $x \in S$ . In order to solve the agnostic learning problem, it is necessary for the learner to understand the how costly these different mistakes are relative to each other. From the learners perspective, observing only the optimal decisions  $y_x$  for each instance  $x$ , such as revealed preferences or expert demonstrations, are clearly insufficient to obtain any information about these costs. One way to overcome this information-theoretic limit of revealed decisions is to directly elicit the utilities from humans – for scenarios  $x$  and decision  $y$ , ask an expert “What is the utility  $u^*(x, y)$  for taking the decision  $y$  given situation  $x$ ?”. However, since the underlying utility  $u^*$  can be quite complex, humans are inept at answering them reliably [17, 23]. For instance, it can be challenging for humans to correctly specify the costs of mispredicting, say, a stop sign as a red signal relative to that of predicting it as a post-box.

On the other hand, it is often easier for humans to provide comparative evaluations based on these utilities [24, 14] and allow the learner to obtain relative feedback. Using these, the learner can query an expert with comparison or preference queries asking “For instance  $x$ , which of the two utilities  $u^*(x, y_1)$  or  $u^*(x, y_2)$  is larger?”. Such vanilla comparisons can allow the learner to infer relative utilities for decisions  $y_1$  and  $y_2$  for a given context  $x$ ; the learner can conclude that mispredicting stop sign as post-box is worse than mispredicting it



as a red signal. However, such feedback still does not provide any information about the mistake costs across different examples – given a choice, should the learner correctly predict a stop-sign or correctly predict a post-box?

While vanilla comparisons are insufficient for the agnostic setup, let us consider the other extreme: suppose that we have access to an oracle which can provide us with comparisons of overall utilities for functions  $f_1, f_2 \in \mathcal{F}$ . That is, the oracle can answer question of the form “Which of the two overall utilities  $\mathbb{E}_x[u^*(x, f_1(x))]$  or  $\mathbb{E}_x[u^*(x, f_2(x))]$  is larger?”. Given access to such an oracle, we will be able to find the optimal classifier in the class  $\mathcal{F}$ . We call this the  $\infty$ -comparison oracle since such preferences requires a human to reason about the utilities over the entire feature space  $\mathcal{X}$  at once. Even for a small image classification task with a million images, this would require a human to compare the utility of a million simultaneous predictions! While this approach does allow for optimal estimation, the trade-off is that it puts the complete burden of learning on the human’s side. It is worth highlighting that the comparisons between lotteries used to establish the von Neumann-Morgenstern utility theorem [18] can be shown to be a special case of such an  $\infty$ -comparison oracle.

While comparison queries only allow comparison within a single instance, the  $\infty$ -comparison oracle takes the other extreme and requires a comparison along all instances. However, we need not restrict our self to either of these extremes; our key insight is that there is a natural spectrum of such comparisons, which we call  $k$ -comparisons which interpolate between the single or 1-comparison and the  $\infty$ -comparison oracle. Such comparison queries allow a learner to pick  $k$  instances  $\{x_1, \dots, x_k\}$  and two sets of corresponding decision,  $\{y_1, \dots, y_k\}$  and  $\{y'_1, \dots, y'_k\}$ , and ask “Which of the cumulative utilities  $\sum_i u^*(x_i, y_i)$  or  $\sum_i u^*(x_i, y'_i)$  is bigger?”. For instance, for the example in Figure 1, giving the learner access to a 2-comparison oracle allows the algorithm to output the optimal decision function.

These higher-order comparison oracles form a natural hierarchy of elicitation mechanisms for the learner with a  $k'$ -oracle being strictly more informative than a  $k$ -oracle for  $k' > k$ . They allow for a natural trade-off between accuracy and elicitation in the learning with unknown utilities framework. As we increase the order  $k$  of the oracle, the learner can obtain finer information about the utilities  $u^*$  and output functions with lower excess risk. However, this increase in information comes at the expense of asking for a harder elicitation from the human expert.

## Our Contributions

We propose a novel framework, which we call *agnostic learning with unknown utilities*, for studying decision problems wherein the learner is evaluated with respect to an unknown utility function. Within this framework, we show that standard approaches which work well in the realizable setup, such as revealed preferences as well as vanilla comparisons, can perform quite poorly in the face of misspecification and can have excess risk  $\Omega(1)$ . To overcome this, we propose a family of elicitation mechanisms, the  $k$ -comparisons, which allows the learner access to finer information from an human expert with increasing values of the order  $k$ . Our main results, detailed in Section 3, provide a tight characterization of the excess risk as a function of the order  $k$  of the comparison oracle available to the learner. These result brings out an interesting accuracy-elicitation trade-off – as the order  $k$  of the oracle increases, the comparative queries allow for more accurate learning in our setup but become harder to elicit from humans.

We would like to highlight that increasing the order  $k$  of the comparisons could lead to potentially biased and noisy responses from the human expert. As a consequence, there might be an additional trade-off involving the *quality of the information* obtained by increasing the order. While we do not focus on this aspect of elicitation, it is an interesting direction for future work.

## Paper Organization

The remainder of the paper is organized as follows: Section 2 introduces our agnostic learning with unknown utilities problem setup and the  $k$ -comparison elicitation mechanism, and Section 3 gives an overview of our main results and algorithmic contributions. In Section 4, we study excess risk bounds for the binary decision problem in our framework and propose our algorithm, Comptron, to learn from higher-order comparisons and in Section 5, we study adaptive estimators which are optimal for each instance of our problem. While these sections contain a formal statement of all our main results, due to space limitations, we defer some of the proofs to full version of the paper.

## 2 Problem formulation

In this section, we formally state our learning with unknown utilities problem and introduce the  $k$ -comparison oracle. Let  $\mathcal{X} \subseteq \mathbb{R}^d$  represent the space of feature vectors,  $\mathcal{Y}$  denote the corresponding decision space and  $\mathcal{F}$  denote a class of decision making functions, given as  $\mathcal{F} = \{f \mid f : \mathcal{X} \mapsto \mathcal{Y}\}$ . Our framework considers an underlying utility function  $u^* : \mathcal{X} \times \mathcal{Y} \mapsto [0, 1]$  which assigns a non-negative real value for making a decision  $y \in \mathcal{Y}$  given a situation  $x \in \mathcal{X}$ . Further, let us denote the set

$$\mathcal{U} = \{u \mid u : \mathcal{X} \times \mathcal{Y} \mapsto [0, 1]\} \quad (3)$$

of all possible such utility functions. For any distribution  $\mathcal{D}_x$  over the feature space  $\mathcal{X}$ , we define the expected utility of a decision function  $f \in \mathcal{F}$  as  $U(f; u^*) := \mathbb{E}_{x \sim \mathcal{D}_x}[u^*(x, f(x))]$ . Observe that such an expected utility model assumes that the utilities are additive across the different instances  $x$  and is a commonly studied model both in the machine learning, statistics and economics literature. We denote the excess risk of a function  $f$  with respect to the function class  $\mathcal{F}$  by

$$\text{err}(f, \mathcal{F}; u^*) := \max_{f' \in \mathcal{F}} U(f'; u^*) - U(f; u^*). \quad (4)$$

Further, we denote the optimal decision for any instance  $x$  with respect to the underlying utility  $u^*$  by  $y_x := \operatorname{argmax}_{y \in \mathcal{Y}} u^*(x, y)$ .

Similar to the classical agnostic learning setup [15], we assume that the learner does not know the underlying distribution  $\mathcal{D}_x$  of the instances. However, our setup differs from it in that we do not assume that the underlying utility function  $u^*$  is known to the learner. Instead, we provide the learner access to an oracle which allows the learner to elicit responses to higher-order preferences queries.

### Comparison Oracle

Since the utility function  $u^*$  is unknown to the learner, our framework allows the learner access to an oracle which provides comparative feedback based on the utilities  $u^*$ . We consider a family of such oracles  $\mathcal{O}_k$ , each indexed by its order  $k$  which determines the number of different instances the learner is allowed to specify in the comparison query. For an oracle  $\mathcal{O}_k$ , a learner is allowed to select a set of  $k$  situations  $\mathbf{x} \in \mathcal{X}^k$  and two pairs of corresponding decisions  $\mathbf{y}_1, \mathbf{y}_2 \in \mathcal{Y}^k$ . The oracle then compares, in a possibly noisy manner, the cumulative utilities of the pair  $(\mathbf{x}, \mathbf{y}_1)$  and  $(\mathbf{x}, \mathbf{y}_2)$  and responds with the feedback on which one is larger. As the order  $k$  of the oracle increases, the queries become more complex – an expert is required to evaluate a larger number of instances at once. This family of comparison oracles captures a natural hierarchy of elicitation mechanisms where with each increasing value of  $k$ , a learner has access to more information about the utility function  $u^*$ .

Formally, we represent a  $k$ -query by a tuple  $(\mathbf{x}, \mathbf{y}_1, \mathbf{y}_2)$  where the input  $\mathbf{x} = (x_1, \dots, x_k)$  comprises  $k$  feature vectors and the corresponding decision vectors  $\mathbf{y}_1 = (y_1, \dots, y_k)$  and  $\mathbf{y}_2 = (y'_1, \dots, y'_k)$ .<sup>3</sup> Given such a query  $q$ , the oracle  $\mathcal{O}_k$  provides the learner a binary response

$$\mathcal{O}_k(q = (\mathbf{x}, \mathbf{y}_1, \mathbf{y}_2)) = \begin{cases} \mathbb{I}[u^*(\mathbf{x}, \mathbf{y}_1) \geq u^*(\mathbf{x}, \mathbf{y}_2)] & \text{with prob. } 1 - \eta_q, \\ 1 - \mathbb{I}[u^*(\mathbf{x}, \mathbf{y}_1) \geq u^*(\mathbf{x}, \mathbf{y}_2)] & \text{otherwise} \end{cases}, \quad (5)$$

where the parameter  $0 \leq \eta_q < \frac{1}{2}$  represents the noise level corresponding to query  $q$ . Thus, the oracle<sup>4</sup>  $\mathcal{O}_k$  provides noisy comparisons of the cumulative utilities  $u^*(\mathbf{x}, \mathbf{y}_1)$  and  $u^*(\mathbf{x}, \mathbf{y}_2)$  with varying noise level  $\eta_q$ . Observe that we allow the noise levels  $\eta_q$  to be different for each query  $q$ .

### Problem Statement

We are interested in the *agnostic learning with unknown utilities* problem where a learner is provided  $n$  samples  $S = \{x_1, \dots, x_n\}$  with each  $x_i \sim \mathcal{D}_x$  and access to the  $k$ -comparison oracle described above, and is required to output a decision function  $\hat{f} \in \mathcal{F}$  such that error  $\text{err}(\hat{f}, \mathcal{F})$  is small. The caveat is to do so with a minimum number of calls, which we term the query complexity  $n_q$  of learning, to the comparison oracle  $\mathcal{O}_k$ . Quantitatively, we would like to characterize the excess risk from equation (4) in terms of the number of sampled instances  $n$ , the order  $k$  of the comparison oracle and properties of the decision function class  $\mathcal{F}$ , and the associated oracle query complexity  $n_q$  to obtain this bound.

Obtaining such bounds on the excess risk  $\text{err}(f, \mathcal{F}; u^*)$  in terms of the order  $k$  allow us to quantify the trade-offs in learning better decision functions at the expense of requiring more complex information from the human expert. Going forward, we focus on the binary decision making problem where the label space  $\mathcal{Y} = \{0, 1\}$  for clarity of exposition. Whenever our results can be extended to arbitrary decision sets, we provide a small remark about this extension.

## 3 Main results

With the formal problem setup in place, we discuss our main results for learning in this framework of unknown utilities. At a high level, our objective is to understand how the excess risk  $\text{err}(f, \mathcal{F}; u^*)$  defined in equation (4) behaves as a function of the oracle order  $k$  – specifically, at what rates does learning in our proposed framework get easier as we allow learner to elicit more complex information from the oracle?

For our main results, on the upper bound side, we design estimators for learning from the  $k$ -comparison oracle, and on the lower bound side, we study information-theoretic limits of learning with such higher-order comparisons. While we state our results for the binary decision problem where the label space  $\mathcal{Y} = \{0, 1\}$  for clarity, most of our results can be generalized to arbitrary outcome space  $\mathcal{Y}$ .

<sup>3</sup> We overload our notation and represent the cumulative utilities of the  $k$  inputs  $(\mathbf{x}, \mathbf{y})$  by  $u^*(\mathbf{x}, \mathbf{y}) = \sum_i u^*(x_i, y_i)$ .

<sup>4</sup> Note that while the oracle depends on the underlying utility function  $u^*$ , our notation suppresses this dependence for clarity. We use the notation  $\mathcal{O}_k(q; u^*)$  whenever we want to make this dependence explicit.

### 3.1 Excess risk with $k$ -comparison oracle (Section 4)

We study a class of *plug-in* estimators which are based on the following two-step procedure:

- i. Obtain estimate  $\hat{u}$  of the true utility  $u^*$  on the sampled datapoints.
- ii. Output utility maximizing function  $\hat{f}_{k,n}$  with respect to the estimated utility  $\hat{u}$ .

For learning the parameters  $\hat{u}$ , we introduce the Comptron (Algorithm 1) and Rob-Comptron (Algorithm 2) algorithms for the noiseless and noisy comparison oracles respectively. We show that when these estimates  $\hat{u}$  are combined with the two-step plug-in estimator, the excess risk of the function  $\hat{f}_{k,n}$  scales as  $O(\frac{1}{k})$  and an additive complexity term capturing uniform convergence of the decision class  $\mathcal{F}$  with respect to the true utility  $u^*$ .

► **Theorem 1** (Informal, noiseless comparisons). *Given  $n$  samples, the excess risk for the function  $\hat{f}_{k,n} \in \mathcal{F}$  output by the plug-in estimator using estimates  $\hat{u}$  from Comptron satisfies*

$$\text{err}(\hat{f}_{k,n}, \mathcal{F}; u^*) \leq \text{Complexity}_n(\mathcal{F}; u^*) + O\left(\frac{1}{k}\right) \cdot \left(\frac{1}{n} \sum_{i=1}^n \mathbb{I}[f_{\text{ERM}}(x_i) \neq y_i]\right),$$

where the ERM function  $f_{\text{ERM}} \in \arg\max_{f \in \mathcal{F}} \sum_{i=1}^n u^*(x_i, f(x_i))$ . Furthermore, Comptron makes only  $O(n \log k)$  queries to the oracle  $\mathcal{O}_k$ .

We make a few remarks on this result. First, observe that the complexity term depends on the true utility function  $u^*$  and not on the estimates  $\hat{u}$ . This ensures that the complexity term does not depend on the utility class  $\mathcal{U}$  but rather only on the specific utility  $u^*$  – indeed, the class  $\mathcal{U}$  consists of all bounded function and uniform convergence might not even be possible with finite sample for a large class of distributions  $\mathcal{D}_x$ . Second, the additional error of  $O(\frac{1}{k})$  accounts for the fact that the utilities  $u^*$  are unknown. One can learn better decision functions by increasing the order  $k$  of the comparison oracle but this comes at the cost of the human expert answering a more complex set of queries. Furthermore, this error is multiplied by the 0 – 1 prediction error of the optimal on-sample classifier  $f_{\text{ERM}} = \arg\max_{f \in \mathcal{F}} \sum_i u^*(x_i, f(x_i))$ . This implies that in the well-specified setup, where there exists an  $f \in \mathcal{F}$  such that  $f(x_i) = y_i$  on the sampled datapoints, the second term becomes 0 and the learner pays no additional error for not knowing the utilities  $u^*$ . Third, observe that our proposed algorithms, Comptron and Rob-Comptron, are query efficient; both require only  $O(n \log k)$  calls to the  $k$ -comparison oracle to produce “good” estimates  $\hat{u}$ .

The proof of the above theorem proceeds in two steps. First, we adapt the classical proof for upper bounding the risk of ERM procedures to show that the gap  $\text{err}(\hat{f}_{k,n}, \mathcal{F})$  decomposes into the complexity term and estimation error  $\|\hat{u} - u^*\|_{S, \infty}$ , evaluated on the dataset  $S$ . Next, we show that this estimation error scales as  $O(\frac{1}{k})$  for the Comptron and Rob-Comptron procedures.

Next, we address the optimality of the above plug-in procedure by studying the information-theoretic limits of learning with a  $k$ -comparison oracle. Specifically, in Theorem 9 we establish that the rate of  $\frac{1}{k}$  is indeed minimax optimal – for any  $k > 1$  and any predictor  $\hat{f}$  in some class  $\mathcal{F}$ , we can construct utility functions  $u^*$  such that excess risk  $\text{err}(\hat{f}, \mathcal{F}; u^*) = \Omega(\frac{1}{k})$ . These lower bounds imply that traditional comparison based learning, corresponding to  $k = 1$ , is insufficient for learning good decision rules in our framework.

### 3.2 Instance-optimal learning (Section 5)

While the previous results show that the error rate of  $O(\frac{1}{k})$  is optimal on worst-case instances, some instances of our learning with unknown utilities problem might be easier than these worst-case ones and one would expect the excess risk to be smaller for them. In this section, we study estimators whose error adapts to hardness of the specific problem instance.

To begin with, in Proposition 11 we establish that the plug-in estimator with Comptron estimates  $\hat{u}$  is not optimal for all instances – it does not adapt to these easier instances. Inspired from the robust optimization literature, we introduce a randomized estimator  $p_{\text{rob}}$  and show that it is instance-optimal. Informally, we establish in Theorem 13 that for any instance  $(\mathcal{D}_x, u^*, \mathcal{F})$  of the problem, the excess risk for  $p_{\text{rob}}$  is characterized by a local modulus of continuity; this modulus captures how quickly the optimal decision function in class  $\mathcal{F}$  can change in a small neighborhood around  $u^*$  for the distribution  $\mathcal{D}_x$ . In Theorem 12, we derive a lower bound on the *local minimax excess risk* and show that the local modulus is indeed the correct instance-dependent complexity measure for this problem.

However, note that such adaptivity to the hardness of the instance comes at the cost of query efficiency. Our estimator  $p_{\text{rob}}$  makes an exponential number  $O(n^k)$  of calls to the oracle  $\mathcal{O}_k$ .

## 4 Binary decision-making with $k$ -comparisons

In this section, we obtain upper and lower bounds on the excess risk for the binary prediction problem with unknown utilities where the learner can elicit utility information using a  $k$ -comparison oracle. In Section 4.1, we introduce algorithms which learn decision-making rules from higher-order preference queries and obtain upper bounds on the excess risk for such estimators. Then, in Section 4.2, we turn to the information-theoretic limits of learning from  $k$ -queries and obtain lower bounds on the minimax risk of any estimator.

Recall from Section 2, our setup gives the learner access to a dataset  $S = \{x_1, \dots, x_n\}$  comprising  $n$  points, each sampled i.i.d. from an underlying distribution  $\mathcal{D}_x$  and to a comparison oracle  $\mathcal{O}_k$ . Before proceeding to define the estimator, we introduce some notation. For any function  $f \in \mathcal{F}$ , let us denote the empirical cumulative utility with respect to utility function  $u^*$  and the corresponding empirical utility maximizer as

$$\hat{U}_n(f; u^*) = \frac{1}{n} \sum_i u^*(x_i, f(x_i)) \quad \text{and} \quad f_{\text{ERM}} \in \operatorname{argmax}_{f \in \mathcal{F}} \hat{U}_n(f; u^*), \quad (6)$$

where the subscript  $n$  encodes the dependence on the number of samples. If the underlying utility  $u^*$  were in fact known to the learner, it would have output the classifier  $f_{\text{ERM}}$ , which, from the classical learning theory literature, is known to have favorable generalization properties [22]. For the case of unknown utilities, we extend this ERM procedure to a natural two-stage plug-in estimator which outputs the minimizer with respect to an estimate  $\hat{u}_k$  of these utilities.

### 4.1 Excess-risk upper bounds for plug-in estimator

Building on the ERM estimator  $f_{\text{ERM}}$  described in equation (6), we design a two stage *plug-in* estimator  $\hat{f}_{k,n}$ , where the subscript  $k$  represents the order of the comparison oracle used to obtain the estimate.

In the first stage, we form estimates  $\hat{u}_k$  of the true utility function  $u^*$  on the sampled datapoints  $S$  using the  $k$ -comparison oracle. The predictor  $\hat{f}_{k,n} \in \mathcal{F}$  is then given by the empirical utility maximizer with respect to  $\hat{u}_k$ , that is,

$$\hat{f}_{k,n} \in \operatorname{argmax}_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^n \hat{u}_k(x_i, f(x_i)). \quad (7)$$

Before detailing out the procedures for producing utility estimates  $\hat{u}_k$ , we present our first main result which shows that the excess risk  $\text{err}(\hat{f}_{k,n}, \mathcal{F}; u^*)$  can be upper bounded as a sum of two terms: (i) a complexity term corresponding to the rate of uniform convergence of the

cumulative utility  $U(f; u^*)$  over the decision class  $\mathcal{F}$  and (ii) an estimation error term which denotes how well the estimates  $\hat{u}_k$  approximate  $u^*$  on the sampled datapoints. Our result measures this estimation error in terms of a data-dependent norm

$$\|u\|_{S, \infty} := \sup_{i \in [n]} \sup_{y \in \mathcal{Y}} |u(x_i, y)|. \quad (8)$$

Recall from equation (6) that the function  $f_{\text{ERM}}$  is the minimizer of the empirical utility  $\hat{U}_n(f; u^*)$ . While the following results hold for general decision spaces  $\mathcal{Y}$ , we later specialize this in Proposition 3 for the binary prediction setup.

► **Theorem 2** (Excess-risk upper bound). *Given datapoints  $S = \{x_1, \dots, x_n\}$  such that each  $x_i \sim \mathcal{D}_x$ , and an estimate  $\hat{u}_k$  of the true utility function  $u^*$ , the plug-in estimate  $\hat{f}_{k,n}$  from equation (7) satisfies*

$$\text{err}(\hat{f}_{k,n}, \mathcal{F}; u^*) \leq 2 \cdot \sup_{f \in \mathcal{F}} (|U(f; u^*) - \hat{U}_n(f; u^*)|) + 2\|u^* - \hat{u}_k\|_{S, \infty} \cdot \left( \frac{1}{n} \sum_{i=1}^n \mathbb{I}[f_{\text{ERM}}(x_i) \neq \hat{f}_{k,n}(x_i)] \right). \quad (9)$$

A few comments on Theorem 2 are in order. First, notice that the upper bound on the risk  $\text{err}(\hat{f}_{k,n}, \mathcal{F}; u^*)$  is a deterministic bound comprising two terms. The uniform convergence term captures how fast the empirical utility  $\hat{U}_n(f; u^*)$  converge to the population utility  $U(f; u^*)$  uniformly over the decision class  $\mathcal{F}$ . Using standard bounds [8], one can show that this term is upper bounded by the empirical Rademacher complexity of the class  $\mathcal{F}$  on the datapoints  $S$ , that is,

$$\sup_{f \in \mathcal{F}} (|U(f; u^*) - \hat{U}_n(f; u^*)|) \leq \mathbb{E}_\varepsilon \left[ \sup_{f \in \mathcal{F}} \left| \frac{1}{n} \sum_{i=1}^n \varepsilon_i u^*(x_i, f(x_i)) \right| \right] := \hat{\mathfrak{R}}_n(\mathcal{F} \circ u^*) \quad (10)$$

where each  $\varepsilon_i$  is an i.i.d. Rademacher random variable taking values  $\{-1, +1\}$  equiprobably. Such complexity measures are commonly studied in the learning theory literature and one can obtain sample complexity rates for a wide range of decision classes including parametric decision classes and non-parametric kernel classes amongst others.

The second term in equation (9) is given by a product of two terms. The first part  $\|u^* - \hat{u}_k\|_{S, \infty}$  captures the *on-sample approximation error* of the estimates  $\hat{u}_k$ . Notice that, in general, the problem of estimating  $u^*$  uniformly over the space  $\mathcal{X}$  is infeasible since the class  $\mathcal{U}$  contains the set of all bounded functions on  $\mathcal{X} \times \mathcal{Y}$ . However, the fact that we are required to estimate the utilities  $u^*$  only on the sampled datapoints  $S$  makes learning feasible in our framework. The second part,  $\frac{1}{n} \sum_{i=1}^n \mathbb{I}[f_{\text{ERM}}(x_i) \neq \hat{f}_{k,n}(x_i)] \leq 1$  the mismatch between the predictions of  $f_{\text{ERM}}$ , obtained with complete knowledge of  $u^*$ , and of  $\hat{f}_{k,n}$ , obtained from estimates  $\hat{u}_k$ . Notice that whenever the function class  $\mathcal{F}$  is correctly specified on  $S$ , that is, there exists a function  $f \in \mathcal{F}$  such that  $f(x_i) = y_i$ , then the predictions of  $\hat{f}_{k,n}$  and  $f_{\text{ERM}}$  will coincide. This follows since the labels  $y_i$  can be inferred using a 1-comparison. In such a well-specified setup, this second term vanishes and we recover the upper bound in terms of the uniform convergence term. Surprisingly, this exhibits that not knowing the utility  $u^*$  affects learnability only when the function class  $\mathcal{F}$  is misspecified.

**Proof.** We begin by decomposing the excess error  $\text{err}(\hat{f}_{k,n}, \mathcal{F}; u^*)$  and then handle each term in the decomposition separately. Recall that the function  $f_{\text{ERM}}$  is the maximizer of the empirical utility  $\hat{U}_n(f; u^*)$ . Then, for any decision function  $f \in \mathcal{F}$ , consider the error

$$\begin{aligned}
\text{err}(\hat{f}_{k,n}, f; u^*) &= U(f; u^*) - \hat{U}_n(f; u^*) + \hat{U}_n(f; u^*) - \hat{U}_n(f_{\text{ERM}}; u^*) + \hat{U}_n(f_{\text{ERM}}; u^*) - \hat{U}_n(\hat{f}_{k,n}; u^*) \\
&\quad + \hat{U}_n(\hat{f}_{k,n}; u^*) - U(\hat{f}_{k,n}; u^*) \\
&\stackrel{(i)}{\leq} 2 \sup_{f \in \mathcal{F}} (|U(f; u^*) - \hat{U}_n(f; u^*)|) + \underbrace{\hat{U}_n(f_{\text{ERM}}; u^*) - \hat{U}_n(\hat{f}_{k,n}; u^*)}_{\text{Term (I)}}, \tag{11}
\end{aligned}$$

where the inequality (i) follows by noting that  $f_{\text{ERM}}$  is the maximizer of  $\hat{U}_n(f; u^*)$ . We now focus our attention on Term (I) in the above expression.

$$\begin{aligned}
\hat{U}_n(f_{\text{ERM}}; u^*) - \hat{U}_n(\hat{f}_{k,n}; u^*) &= \hat{U}_n(f_{\text{ERM}}; u^*) - \hat{U}_n(f_{\text{ERM}}; \hat{u}) + \hat{U}_n(f_{\text{ERM}}; \hat{u}) - \hat{U}_n(\hat{f}_{k,n}; \hat{u}) \\
&\quad + \hat{U}_n(\hat{f}_{k,n}; \hat{u}) - \hat{U}_n(\hat{f}_{k,n}; u^*) \\
&\stackrel{(i)}{\leq} \frac{2}{n} \sum_{i=1}^n \mathbb{I}[f_{\text{ERM}}(x_i) \neq \hat{f}_{k,n}(x_i)] \cdot \sup_{y \in \mathcal{Y}} |u^*(x_i, y) - \hat{u}(x_i, y)| \\
&\leq 2 \|u^* - \hat{u}\|_{S, \infty} \cdot \left( \frac{1}{n} \sum_{i=1}^n \mathbb{I}[f_{\text{ERM}}(x_i) \neq \hat{f}_{k,n}(x_i)] \right),
\end{aligned}$$

where (i) follows by noting that  $\hat{f}_{k,n}$  maximizes the utility  $\hat{U}_n(f; \hat{u})$ . Plugging the bound above in equation (11) completes the proof.  $\blacktriangleleft$

We now specialize the result of Theorem 2 to the binary prediction setup where the label space  $\mathcal{Y} = \{0, 1\}$ . Recall that for each datapoint  $x_i$ , we denote the true label by  $y_i = \arg\max_y u^*(x_i, y)$ . We now introduce the notion of utility gaps  $u_{\text{gap}}(x_i)$  which measures the excess utility a learner gains by predicting a datapoint  $x_i$  correctly relative to an incorrect prediction. Formally, the gap  $u_{\text{gap}}(x_i)$  for datapoint  $x_i$  with respect to some utility function  $u \in \mathcal{U}$  is given as

$$u_{\text{gap}}(x_i) := u(x_i, y_i) - u(x_i, \bar{y}_i), \tag{12}$$

where we denote the incorrect label by  $\bar{y} = 1 - y$ . With this notation, the following proposition obtains an upper bound on the excess error of plug-in estimator  $\hat{f}_{k,n}$  for the binary prediction problem in terms of the estimation error in these gaps  $u_{\text{gap}}(x_i)$ .

► **Proposition 3** (Upper bounds for binary prediction). *Consider the binary decision making problem with label space  $\mathcal{Y} = \{0, 1\}$ . Given  $n$  datapoints  $\{x_1, \dots, x_n\}$  such that each datapoint  $x_i \sim \mathcal{D}_x$ , and an estimate  $\hat{u}_k$  of the utility function  $u^*$ , the plug-in estimator  $\hat{f}_{k,n}$  from equation (7) satisfies*

$$\begin{aligned}
\text{err}(\hat{f}_{k,n}, \mathcal{F}; u^*) &\leq 2 \cdot \sup_{f \in \mathcal{F}} (|U(f; u^*) - \hat{U}(f; u^*)|) \\
&\quad + 2 \max_i [u_{\text{gap}}^*(x_i) - \hat{u}_{\text{gap}}(x_i)] \cdot \left( \frac{1}{n} \sum_{i=1}^n \mathbb{I}[f_{\text{ERM}}(x_i) \neq y_i] \right). \tag{13}
\end{aligned}$$

The proof of the above proposition follows similar to Theorem 2 and is deferred to the full version. This specializes the result of Theorem 2 and shows that for the binary prediction problem, estimating the utility gaps  $u_{\text{gap}}$  well for each datapoint suffices

The upper bound on excess risk given by Proposition 3 shows that the function  $\hat{f}_{k,n}$  derived from estimates  $\hat{u}_k$  will have small error as long as the estimates  $\hat{u}_{\text{gap}}(x_i)$  approximate the true utility gaps  $u_{\text{gap}}^*(x_i)$  for each datapoint  $x_i$ . Therefore, in the following sections, we focus on procedures for obtaining the utility estimates  $\hat{u}_{\text{gap}}$  using the  $k$ -comparison oracle. we separate the presentation based on whether the oracle  $\mathcal{O}_k$  provides noiseless comparisons ( $\eta_q = 0$  for all  $q$ ) or whether the oracle evaluations are noisy.



■ **Algorithm 1** Comptron: Comparison based Coordinate-Perceptron for estimating  $u_{\text{gap}}^*$ .

---

**Input:** Datapoints  $S = \{x_1, \dots, x_n\}$ ,  $k$ -comparison oracle  $\mathcal{O}_k$   
**Initialize:** Set  $T = \log_2 k - 1$   
 Obtain  $y_i = \operatorname{argmax}_y u^*(x_i, y)$  for each  $i$  using 1-comparison.  
 Obtain index  $i_{\max}$  using 2-comparisons such that  $i_{\max} = \operatorname{argmax}_i u_{\text{gap}}^*(x_i)$ .  
 Set initial estimates  $\hat{u}_{\text{gap}}^0 = [\hat{u}_{\text{gap}}^0(x_1), \dots, \hat{u}_{\text{gap}}^0(x_n)] = u_{\max}^* := u_{\text{gap}}^*(x_{i_{\max}})$ .  
 (Note that exact value of  $u_{\max}^*$  is not required since comparison queries are relative)  
**for**  $t = 1, \dots, T$  **do**  
   **for**  $i = 1, \dots, n$  **do**  
     Denote by  $\lambda = \frac{k}{2u_{\max}^*} \left( \hat{u}_{\text{gap}}^{t-1}(x_i) - \frac{u_{\max}^*}{2^t} \right)$  and query  $q_{i,t} = (\mathbf{x}, \mathbf{y}_1, \mathbf{y}_2)$  where  
        $\mathbf{x} = (\underbrace{x_i, \dots, x_i}_{\frac{k}{2} \text{ times}}, \underbrace{x_{i_{\max}}, \dots, x_{i_{\max}}}_{\lambda \text{ times}}), \quad \mathbf{y}_1 = (\underbrace{y_i, \dots, y_i}_{\frac{k}{2} \text{ times}}, \underbrace{1 - y_{i_{\max}}, \dots, 1 - y_{i_{\max}}}_{\lambda \text{ times}}), \quad \mathbf{y}_2 = 1 - \mathbf{y}_1$ .  
       Query oracle  $\mathcal{O}_k$  with  $q_{i,t}$  and receive response  $r_{i,t}$ .  
       Update  $\hat{u}_{\text{gap}}^t(x_i) = \hat{u}_{\text{gap}}^{t-1}(x_i) - \mathbb{I}[r_{i,t} = 0] \cdot \frac{u_{\max}^*}{2^t}$ .  
**Output:** Gap estimates  $\hat{u}_{\text{gap}}^T$

---

#### 4.1.1 Estimating $u_{\text{gap}}^*$ with noiseless oracle

In this section, we propose our algorithm for estimating the gaps  $u_{\text{gap}}^*$  when the  $k$ -comparison oracle is noiseless. Recall from equation (5), for a query  $q = (\mathbf{x}, \mathbf{y}_1, \mathbf{y}_2)$  comprising  $k$  feature vectors  $\mathbf{x} = (x_1, \dots, x_k)$ , and two decision vectors  $\mathbf{y}_1 = (y_1, \dots, y_k)$  and  $\mathbf{y}_2 = (y'_1, \dots, y'_k)$ , such a noiseless oracle deterministically outputs

$$\mathcal{O}_k(q = (\mathbf{x}, \mathbf{y}_1, \mathbf{y}_2)) = \mathbb{I}[u^*(\mathbf{x}, \mathbf{y}_1) \geq u^*(\mathbf{x}, \mathbf{y}_2)] ,$$

where recall that  $u^*(\mathbf{x}, \mathbf{y}) = \sum_{i \in [k]} u^*(x_i, y_i)$  is the sum of the utilities under  $u^*$  for the tuple  $(\mathbf{x}, \mathbf{y})$ . In the binary prediction setup, such queries allow a learner to specify a set of  $k$  instances  $\mathbf{x}$  and a subset  $S_q \subset \mathbf{x}$  and ask the oracle “whether correctly predicting instances in  $S_q$  has higher utility or the instances in the complement  $\mathbf{x} \setminus S_q$ ?”

Recall that Proposition 3 shows that excess risk for the plug-in estimator can be bounded by the worst-error  $|u_{\text{gap}}^*(x_i) - \hat{u}_{\text{gap}}(x_i)|$  over the set of sampled datapoints  $S$ . To obtain such estimates, we introduce *Comptron* in Algorithm 1 which is a coordinate-wise variant of the classical perceptron algorithm [20]. At a high level, Comptron is an iterative procedure which estimates the utility gaps  $u_{\text{gap}}^*(x_i)$  for each  $x_i$  relative to the largest gap

$$u_{\max}^* := \max_{i \in [n]} u_{\text{gap}}^*(x_i) \leq 1. \tag{14}$$

At each iteration  $t$ , the queries  $q_{i,t}$  are selected such that  $\hat{u}_{\text{gap}}^{t-1}(\mathbf{x}, \mathbf{y}_1) > \hat{u}_{\text{gap}}^{t-1}(\mathbf{x}, \mathbf{y}_2)$  under the current estimates  $\hat{u}_{\text{gap}}^{t-1}$ . If the oracle’s response is  $r_{i,t} = 1$ , the estimates are consistent with the response and it keeps the current estimate. On the other hand, if the response  $r_{i,t} = 0$ , the algorithm decreases its current estimate of the  $i^{\text{th}}$  datapoint in order to be consistent with this query. Comptron repeats the above procedure for  $T = \log_2 k - 1$  timesteps and finally outputs the estimates  $\hat{u}_{\text{gap}}^T$ .

It is worth highlighting here that Comptron initializes all the estimates as the largest gap, that is,  $\hat{u}_{\text{gap}}^0(x_i) = u_{\max}^*$ . Such an initialization is purely symbolic in nature and the algorithm *does not* require knowledge of this value. This is because the comparison queries  $q_{i,t}$  allows

the algorithm to compare the estimates  $\hat{u}_{\text{gap}}$  with  $u_{\text{max}}^*$  and the algorithm maintains its estimates  $\hat{u}_{\text{gap}}^t$  as a multiplicative factor of  $u_{\text{max}}^*$  for iterations  $t$ . Further, we can use symbolic estimates to output the plug-in estimator since it is invariant to scaling the utility gaps by a positive constant,

$$\begin{aligned} \operatorname{argmax}_{f \in \mathcal{F}} \sum_{i=1}^n \hat{u}(x_i, f(x_i)) &\equiv \operatorname{argmax}_{f \in \mathcal{F}} \sum_{i=1}^n \hat{u}_{\text{gap}}(x_i) \cdot \mathbb{I}[f(x_i) = y_i] \\ &\equiv \operatorname{argmax}_{f \in \mathcal{F}} \sum_{i=1}^n \frac{\hat{u}_{\text{gap}}(x_i)}{u_{\text{max}}^*} \cdot \mathbb{I}[f(x_i) = y_i]. \end{aligned}$$

The following lemma provides an upper bound on the estimation error of Comptron and shows that the output estimates  $\hat{u}_{\text{gap}}(x_i)$  are within a factor  $O(\frac{u_{\text{max}}^*}{k})$  of the true gaps  $u_{\text{gap}}^*(x_i)$ .

► **Lemma 4** (Estimation error of Algorithm 1). *Given access to datapoints  $S = \{x_1, \dots, x_n\}$  and  $k$ -comparison oracle  $\mathcal{O}_k$ , Comptron (Algorithm 1) uses  $O(n \log k)$  queries to the oracle and produces estimates  $\hat{u}_{\text{gap}}$  such that*

$$\max_{i \in [n]} |\hat{u}_{\text{gap}}(x_i) - u_{\text{gap}}^*(x_i)| \leq \frac{2u_{\text{max}}^*}{k}. \quad (15)$$

We defer the proof of the lemma to the full version. The proof proceed via an inductive argument where we show that the confidence interval around  $u_{\text{gap}}^*(x_i)$  shrinks by a factor of  $\frac{1}{2}$  in each iteration for every datapoint  $x_i$ . Given the above estimation error guarantee for Comptron, the following corollary combines these with the excess risk bounds of Proposition 3 to obtain an upper bound on the excess risk of  $\hat{f}_{k,n}$ .

► **Corollary 5.** *Consider the binary decision making problem with label space  $\mathcal{Y} = \{0, 1\}$ . Given  $n$  datapoints  $\{x_1, \dots, x_n\}$  such that each  $x_i \sim \mathcal{D}_x$ , the plug-in estimate  $\hat{f}_{k,n}$  from equation (7), when instantiated with the output of Comptron (Algorithm 1), satisfies*

$$\operatorname{err}(\hat{f}_{k,n}, \mathcal{F}; u^*) \leq 2 \cdot \sup_{f \in \mathcal{F}} (|U(f; u^*) - \hat{U}(f; u^*)|) + \frac{2u_{\text{max}}^*}{k} \cdot \left( \frac{1}{n} \sum_{i=1}^n \mathbb{I}[f_{\text{ERM}}(x_i) \neq y_i] \right).$$

Corollary 5 exhibits the advantage of using higher-order comparisons for the learning with unknown utilities problem – as the order  $k$  increases, the error of the plug-in estimate decreases additively as  $O(\frac{1}{k})$ . It is worth noting here that while the higher-order comparisons allow the learner to better estimate the underlying utilities, the problem gets harder from the side of the human expert. Indeed, with higher values of  $k$ , the expert is required to compare utilities across  $k$  different possible situations which can make the elicitation a harder task.

While the results in this section exhibit how the excess risk  $\operatorname{err}(\hat{f}_{k,n}; \mathcal{F})$  varies as a function of  $k$ , they rely on the oracle responses being noiseless. In the next section, we consider the setup where the oracle responses can be noisy and propose a robust version of the Comptron algorithm for learning in this scenario.

#### 4.1.2 Estimating $u_{\text{gap}}^*$ with noisy oracle

In contrast to the deterministic noiseless oracle of the previous section, here, we consider learning with unknown utilities when the oracle  $\mathcal{O}_k$  can output noisy responses to each query. Recall from equation (5), for any query  $q$ , the noisy  $k$ -comparison oracle the correct response with probability  $1 - \eta_q$  and flips the response with probability  $\eta_q$  for some value of  $\eta_q < \frac{1}{2}$ . While we allow this error probability to vary across different queries, we assume that this error is bounded uniformly across all queries by some constant  $\eta < \frac{1}{2}$ .

■ **Algorithm 2** Rob-Comptron: Robust Comptron for estimating  $u_{\text{gap}}^*$  with noisy oracle.

---

**Input:** Datapoints  $S = \{x_1, \dots, x_n\}$ ,  $k$ -comparison oracle  $\mathcal{O}_k$ , noise level  $\eta$ , confidence  $\delta$

**Initialize:**  $T = \log_2 k - 1$ ,  $J = \frac{8}{(1-2\eta)^2} \log\left(\frac{nT}{\delta}\right)$

Obtain  $y_i = \operatorname{argmax}_y u^*(x_i, y)$  for each  $i$  using 1-comparison.

Obtain index  $i_{\max}$  using 2-comparisons such that  $i_{\max} = \operatorname{argmax}_i u_{\text{gap}}^*(x_i)$ .

Set initial estimates  $\hat{u}_{\text{gap}}^0 = [\hat{u}_{\text{gap}}^0(x_1), \dots, \hat{u}_{\text{gap}}^0(x_n)] = u_{\max}^*$  symbolically

**for**  $t = 1, \dots, T$  **do**

**for**  $i = 1, \dots, n$  **do**

Denote by  $\lambda = \frac{k}{2u_{\max}^*} \left( \hat{u}_{\text{gap}}^{t-1}(x_i) - \frac{u_{\max}^*}{2^t} \right)$

Set query  $q_{i,t} = (\mathbf{x}, \mathbf{y}_1, \mathbf{y}_2)$  where

$\mathbf{x} = (\underbrace{x_i, \dots, x_i}_{\frac{k}{2} \text{ times}}, \underbrace{x_{i_{\max}}, \dots, x_{i_{\max}}}_{\lambda \text{ times}}), \quad \mathbf{y}_1 = (\underbrace{y_i, \dots, y_i}_{\frac{k}{2} \text{ times}}, \underbrace{1 - y_{i_{\max}}, \dots, 1 - y_{i_{\max}}}_{\lambda \text{ times}}), \quad \mathbf{y}_2 = 1 - \mathbf{y}_1.$

**for**  $j = 1, \dots, J$  **do**

Query oracle  $\mathcal{O}_k$  with  $q_{i,t}$  and receive response  $r_{i,j,t}$ .

Update  $\hat{u}_{\text{gap}}^t(x_i) = \hat{u}_{\text{gap}}^{t-1}(x_i) - \mathbb{I}[\frac{1}{J} \sum_j r_{i,j,t} < \frac{1}{2}] \cdot \frac{u_{\max}^*}{2^t}.$

**Output:** Gap estimates  $\hat{u}_{\text{gap}}^T$

---

► **Assumption 6.** For the noisy  $k$ -comparison oracle described in equation (5), we have that  $\eta_q \leq \eta < \frac{1}{2}$  for all queries  $q$ .

From an algorithmic perspective, it is well known that the perceptron algorithm itself is not noise-stable and can oscillate if there are datapoints  $x$  which have noisy labels. In order to overcome this limitation, several noise-robust perceptron variants have been proposed in the literature; see [16] for an extensive review.

We build on this line of work and present Rob-Comptron (Algorithm 2), a noise-robust variant of the deterministic Comptron algorithm. The main difference is the presence of an additional inner-loop with index  $j$  which repeatedly queries  $q_{i,t}$  for  $J = \tilde{O}\left(\frac{1}{(1-2\eta)^2}\right)$  times. In each iteration, the update is again a coordinate-wise perceptron update which matches the prediction of the current estimate with the average of the oracle responses. Such an averaging has been previously used in the context of learning halfspaces from noisy data both in a passive [11] and active [25] framework.

The following lemma provides an upper bound on the estimation error of the gap estimates produced by Rob-Comptron.

► **Lemma 7** (Estimation error of Algorithm 2). Given access to datapoints  $S = \{x_1, \dots, x_n\}$  and noisy  $k$ -comparison oracle  $\mathcal{O}_k$  satisfying Assumption 6 with parameter  $\eta$ , Rob-Comptron (Algorithm 2) uses  $O\left(\frac{n}{(1-2\eta)^2} \cdot \log k \cdot \log \frac{n \log k}{\delta}\right)$  queries and produces estimates  $\hat{u}_{\text{gap}}$  such that

$$\max_{i \in [n]} |\hat{u}_{\text{gap}}(x_i) - u_{\text{gap}}^*(x_i)| \leq \frac{2u_{\max}^*}{k}, \quad (16)$$

with probability at least  $1 - \delta$ .

In comparison to Comptron which requires  $O(n \log k)$  queries to the comparison oracle, the robust variant Rob-Comptron requires a fraction  $\frac{1}{(1-2\eta)^2}$  more queries to achieve a similar estimation error. Such an increase in query complexity is typical of learning with such noisy oracles in the binary classification setup [5, 7, 12, 25].

Similar to Corollary 5 in the previous section, we can combine the above high-probability bound on the estimation error to obtain a bound on the excess risk which scales as  $\frac{1}{k}$  with the order  $k$  of the comparison oracle.

► **Corollary 8.** *Consider the binary decision making problem with label space  $\mathcal{Y} = \{0, 1\}$ . Given  $n$  datapoints  $\{x_1, \dots, x_n\}$  such that each  $x_i \sim \mathcal{D}_x$ , the plug-in estimate  $\hat{f}_{k,n}$  from equation (7), when instantiated with the output of Comptron (Algorithm 1), satisfies*

$$\text{err}(\hat{f}_{k,n}, \mathcal{F}; u^*) \leq 2 \cdot \sup_{f \in \mathcal{F}} \left( |U(f; u^*) - \hat{U}(f; u^*)| \right) + \frac{2u_{\max}^*}{k} \cdot \left( \frac{1}{n} \sum_{i=1}^n \mathbb{I}[f_{\text{ERM}}(x_i) \neq y_i] \right).$$

with probability at least  $1 - \delta$ .

We omit the proof of this corollary since it essentially follows the same steps as that for Corollary 5. This corollary establishes that by increasing the query complexity by a factor of  $O(1/(1-2\eta)^2)$ , one can recover the same additive  $\frac{1}{k}$  excess risk bound of the deterministic setup. Combined, Corollaries 5 and 8 establish the trade-offs in the reduction of the excess risk while eliciting more complex information about the underlying utility  $u^*$  through the  $k$ -comparison oracle.

## 4.2 Information-theoretic lower bounds

In the previous section, we studied the learning with unknown utility problem from an algorithmic perspective and showed that the plug-in estimator with Comptron estimates  $\hat{u}$  achieve an excess risk bound which scales as  $O(\frac{1}{k})$  with the order  $k$  of the comparison. In this section, we ask whether such a scaling of the error term is optimal and study this lower bound question from an information-theoretic perspective.

Recall from Theorem 2 that the excess risk decomposes into two terms: (i) a uniform convergence term for the decision class  $\mathcal{F}$  with respect to utility function  $u^*$  and (ii) an estimation error term corresponding to how well  $\hat{u}_k$  approximates  $u^*$  on the sampled datapoints. When the underlying utility function  $u^*$  is known, classical results from the learning theory literature the uniform convergence complexity term is in general unavoidable [21]. With this, we take the infinite-data limit, where the learner is assumed to have access to the distribution  $\mathcal{D}_x$ , and study whether the excess error of  $O(\frac{1}{k})$  is necessary.

Our notion of minimax risk is based on the subset of utility functions which cannot be distinguished by any learner with access to a  $k$ -comparison oracle. Formally, given any oracle  $\mathcal{O}_k(\cdot; u^*)$ , where we have made the dependence on the utility  $u^*$  explicit, we denote by  $\mathcal{U}_{k,u^*}$  the subset of utility functions in the class  $\mathcal{U}$  which are consistent with the responses of  $\mathcal{O}_k(\cdot; u^*)$ . With this, we define the information-theoretic minimax risk  $\mathfrak{M}_k(\mathcal{F}, \mathcal{D}_x)$  with respect to the function class  $\mathcal{F}$  and distribution  $\mathcal{D}_x$  as

$$\mathfrak{M}_k(\mathcal{F}, \mathcal{D}_x) := \sup_{\mathcal{O}_k(\cdot; u^*)} \inf_{p \in \Delta_{\mathcal{F}}} \sup_{u \in \mathcal{U}_{k,u^*}} \mathbb{E}_{f \sim p} [\text{err}(f, \mathcal{F}; u)], \quad (17)$$

where the infimum is taken over all procedures which take as input the distribution  $\mathcal{D}_x$  over the instances and access to a  $k$ -comparison oracle, and output a possibly randomized estimate  $p \in \Delta_{\mathcal{F}}$ . The above notion of minimax risk can be viewed as a three-stage game

between the learner and the environment. The sequence of supremum and infimum depicts the order in which information is revealed in this game. The environment first selects a  $k$ -query oracle  $\mathcal{O}(\cdot; u^*)$  with underlying utility  $u^*$ . The learner is then provided access to the underlying distribution  $\mathcal{D}_x$ , function class  $\mathcal{F}$  and the oracle  $\mathcal{O}(\cdot; u^*)$  based on which it outputs a possibly randomized decision function given by  $p \in \Delta_{\mathcal{F}}$ . The environment is then allowed to select the worst-case utility  $u$  such that it is consistent with the  $k$ -oracle  $\mathcal{O}(\cdot; u^*)$  and the learner is evaluated in expectation over this chosen utility. We call this the minimax risk of learning with respect to class  $\mathcal{F}$  and distribution  $\mathcal{D}_x$ .

Our next main result shows that there exist instances of the binary prediction problem  $(\mathcal{F}, \mathcal{D}_x)$  such that the minimax risk  $\mathfrak{M}_k(\mathcal{F}, \mathcal{D}_x)$  is lower bounded by  $\frac{1}{k}$  for any  $k \geq 2$  up to some universal constants. Observe that this matches the corresponding upper bounds obtained in Corollaries 5 and 8 exhibiting that the proposed plug-in estimator in equation (7) with Comptron (Rob-Comptron for noisy oracle) utilities is indeed minimax optimal for the binary prediction setup.

► **Theorem 9.** *There exists a universal constant  $c > 0$  such that for any  $k \geq 2$ , there exist a binary prediction problem instance  $(\mathcal{F}, \mathcal{D}_x)$  such that*

$$\mathfrak{M}_k(\mathcal{F}, \mathcal{D}_x) \geq \frac{c}{k}.$$

A few comments on Theorem 9 are in order. First, the above result shows a family of lower bounds for our learning with unknown utilities framework – one for each value of the order  $k$ . Specifically, it shows that for every  $k \geq 2$ , there exists a worst-case instance such that any algorithm will incur an error of  $\Omega(\frac{1}{k})$ . Compare this with the upper bounds on excess risk from the previous section. In the limit of infinite data, Corollaries 5 and 8 exhibit that the excess risk  $\text{err}(\hat{f}_{k,n}, \mathcal{F}; u^*) = O(\frac{1}{k})$  for the plug-in estimator  $\hat{f}_{k,n}$ . This establishes that the plug-in estimator with Comptron and Rob-Comptron utility estimates is indeed minimax optimal.

**Proof.** In order to establish a lower bound on the minimax risk  $\mathfrak{M}_k$ , we will construct two utility functions  $u_1, u_2 \in \mathcal{U}$  such that the  $k$ -comparison oracle has identical responses for both these utility functions. For the purpose of our construction, we will consider noiseless oracle; the problem only becomes harder for the learner if the oracle responses are noisy. Given these two utility functions, we next show that their maximizers  $f_1$  and  $f_2$  are different for some function class  $\mathcal{F}$ . We then combine these two insights to obtain the final minimax bound.

For our lower bound construction, we will focus on a setup where the features are one dimensional with  $\mathcal{X} = \mathbb{R}$  and the linear decision function class

$$\mathcal{F}_{\text{lin}} = \{f_a \mid f_a(x) = \text{sign}(ax), a \in [-1, 1]\}.$$

Recall that for any point  $x$ , we represent by  $u_{\text{gap}}(x) = u(x, y_x) - u(x, \bar{y}_x)$  the utility gain corresponding to the function  $u$ . Before constructing the explicit example, we present a technical lemma which highlights a limitation of a  $k$ -comparison oracle – it establishes that a  $k$ -oracle will not be able to distinguish utility functions for which the utility gaps are in the range  $(1 - \frac{1}{k}, 1)$ .

► **Lemma 10.** *Consider any utility functions  $u_1, u_2 \in \mathcal{U}$ . Let datapoints  $x$  have utility gain  $u_{\text{gap}}^i(x)$  for  $i = \{1, 2\}$ . For any two points  $x_1, x_2$  such that*

$$u_{\text{gap}}^1(x_1) = u_{\text{gap}}^2(x_1) = u_{\text{gap}}(x_1) \quad \text{and} \quad \left(1 - \frac{1}{k}\right) \cdot u_{\text{gap}}(x_1) \leq u_{\text{gap}}^i(x_2) \leq u_{\text{gap}}(x_1),$$

*the oracle responses for any query  $q = (\mathbf{x}, \mathbf{y}_1, \mathbf{y}_2)$  comprising points  $x_1$  and  $x_2$  are identical for  $u^* = u_1$  or  $u^* = u_2$ .*

We defer the proof of the above lemma to the full version. Taking this as given, we proceed with our lower bound construction.

**Utility functions  $u_1$  and  $u_2$ .** Our construction considers two datapoints  $x_+ = +1$  and  $x_- = -1$  and two utility functions  $u$  and  $\tilde{u}$  satisfying

$$\begin{aligned} u_1(x_+, 1) &> u_1(x_+, 0) \quad \text{and} \quad u_1(x_-, 1) > u_1(x_-, 0), \\ u_2(x_+, 1) &> u_2(x_+, 0) \quad \text{and} \quad u_2(x_-, 1) > u_2(x_-, 0). \end{aligned}$$

Observe that under these utilities, any function  $f_a \in \mathcal{F}_{\text{lin}}$  can make a correct decision for either point  $x_+$  or point  $x_-$  but not for both simultaneously. Given these datapoints, the two utility functions are given by

$$\begin{aligned} u_1(x_+, 1) &= 1, \quad u_1(x_-, 1) = 1 - \gamma_1 \quad \text{where} \quad \gamma_1 = \frac{1}{2(3k+1)} \\ u_2(x_+, 1) &= 1, \quad u_2(x_-, 1) = 1 - \gamma_2 \quad \text{where} \quad \gamma_2 = \frac{2}{(3k+1)}, \end{aligned}$$

and  $u_i(x, 0) = 0$  for both  $i = \{1, 2\}$ . Observe that both  $\gamma_1, \gamma_2$  have been set to satisfy the conditions of Lemma 10, that is,

$$\left(1 - \frac{1}{k}\right) \cdot u_{\text{gap}}(x_+) \leq u_{\text{gap}}^i(x_-) \leq u_{\text{gap}}(x_+) \quad \text{for } i = \{1, 2\}.$$

**Distribution  $\mathcal{D}_x$ .** For any  $k > 2$ , consider the distribution  $\mathcal{D}_x$  over the points  $\{x_+, x_-\}$  such that

$$\Pr(x = x_+) = \frac{3k}{6k+1} \quad \text{and} \quad \Pr(x = x_-) = \frac{3k+1}{6k+1}.$$

By Lemma 10, we have that using the  $k$ -comparison oracle, no learner can distinguish between the utility functions  $u_1$  and  $u_2$  on the distribution  $\mathcal{D}_x$ . Further, recall that any classifier  $f_a \in \mathcal{F}_{\text{lin}}$  can either predict  $x_+$  or  $x_-$  correctly. We now obtain a bound on the excess risk  $\text{err}(f_a, \mathcal{F}; u)$  for both these cases separately.

**Case 1:**  $f_a(x_+) = 1$ . In this case, the utility gap is maximized by setting the utility  $u = u_1$  in the minimax risk. The corresponding excess risk is given by

$$\text{err}(f_a, \mathcal{F}; u_1) = \frac{(3k+1)(1-\gamma_1)}{6k+1} - \frac{3k}{6k+1} = \frac{1}{2(6k+1)}. \quad (18)$$

**Case 2:**  $f_a(x_-) = 1$ . In this case, the utility gap is maximized by setting the utility  $u = u_2$  and the excess risk is given by

$$\text{err}(f_a, \mathcal{F}; u_2) = \frac{3k}{6k+1} - \frac{(3k+1)(1-\gamma_2)}{6k+1} = \frac{1}{(6k+1)}. \quad (19)$$

Noting that any predictor  $\hat{f}$  will output a function corresponding to one of the two cases above and combining equations (18) and (19) establishes the desired claim.  $\blacktriangleleft$

While the information theoretic results of this section showed that the plug-in estimator is minimax optimal, the next section focuses on whether this estimator is able to *adapt* to easier problem instances – specifically, whether our estimation procedures Comptron and Rob-Comptron are optimal for every problem instance? We answer this in the negative and introduce a new estimator which is instance optimal. However, such an adaptivity to easier instances comes at the cost of an exponential query complexity.

## 5 Instance-optimal guarantees for binary prediction

In the previous section, we proposed query-efficient algorithms, Comptron and Rob-Comptron, for learning a function  $\hat{f}_{k,n}$  with small excess risk using only  $\tilde{O}(n \log k)$  queries to the  $k$ -comparison oracle. Further, the upper bounds in Corollaries 5 and 8 along with the lower bound of Theorem 9 establish that our proposed algorithms are indeed minimax optimal over the class of utility functions  $\mathcal{U}$ . Given this, it is natural to ask whether our proposed algorithms are instance wise-optimal, that is, do they achieve the best possible excess-risk bounds for *all*  $u^* \in \mathcal{U}$ ?

To simplify our presentation, we study this question at the population level,<sup>5</sup> where we assume that the learner has access to the underlying distribution  $\mathcal{D}_x$ . This allows us to focus on the excess risk as a function of the order  $k$  of the comparison oracle and ignore the uniform convergence term. We also restrict our attention to the deterministic noiseless oracle since one can reduce the noisy oracle to the noiseless oracle by using the averaging technique presented in Section 4.1.

The following proposition shows that the plug-in estimator with Comptron utilities are *not instance-optimal*, that is, it does not adapt to the hardness of the learning with unknown utilities problem instance. Specifically, it constructs a problem instance  $(\mathcal{F}, \mathcal{D}_x)$  with a noiseless oracle and shows that the estimate<sup>6</sup>  $\hat{f}_k$  from equation (7) with Comptron utility estimates has an excess risk of  $\frac{1}{k}$  while there exists an estimator, which uses all  $k$ -queries and is able to achieve zero excess risk.

Recall that for any utility  $u^* \in \mathcal{U}$ , we denote by  $\mathcal{U}_{k,u^*}$  the subset of utility functions in the class  $\mathcal{U}$  which are indistinguishable from  $u^*$  under the  $k$ -comparison oracle  $\mathcal{O}(\cdot; u^*)$ .

► **Proposition 11** (Plug-in with Comptron estimates is not instance-optimal). *For every  $k > 2$ , there exists a binary prediction instance  $(\mathcal{F}, \mathcal{D}_x)$  along with an oracle  $\mathcal{O}_k$  such that*

a) *The error of the plug-in estimate  $\hat{f}_k$  from equation (7) with estimated utilities  $\hat{u}_k$  from Comptron (Algorithm 1) is non-zero, that is,*

$$\text{err}(\hat{f}_k, \mathcal{F}; u^*) = \frac{1}{k}.$$

b) *There exists an optimal predictor  $\tilde{f}$  with zero excess-risk, that is,*

$$\sup_{u \in \mathcal{U}_{k,u^*}} \text{err}(\tilde{f}, \mathcal{F}; u) = 0.$$

We make a few remarks about the proposition. While the first part of the proposition shows that the excess risk  $\text{err}(\hat{f}_k, \mathcal{F}; u^*) = \frac{1}{k}$ , the second part makes a stronger claim about the performance of  $\tilde{f}$  on all utilities  $u \in \mathcal{U}_{k,u^*}$ . This shows that the predictor  $\tilde{f}$  performs well when evaluated on an entire neighborhood around the true utility  $u^*$ .

Having established that our estimators from the previous section are not adaptive, we introduce a notion of *local minimax risk* and study estimators which are instance-optimal. We begin by precisely defining this notion of instance-wise minimax optimality. Recall from Section 4.2, our notion of minimax risk  $\mathfrak{M}_k(\mathcal{F}, \mathcal{D}_x)$  was a worst-case notion – the minimax risk was defined as a supremum over all oracles  $\mathcal{O}_k(\cdot; u^*)$ . We extend this global minimax notion to a local minimax one. In particular, for any  $u^* \in \mathcal{U}$ , we define the local minimax risk around  $u^*$  as

<sup>5</sup> Our analysis could be extended to the finite sample setup using the bound obtained in Theorem 2.

<sup>6</sup> Since we are working at the population level, we have dropped the subscript  $n$  from  $\hat{f}_{k,n}$ .



$$\mathfrak{M}_k(\mathcal{F}, \mathcal{D}_x; u^*) := \inf_{\hat{f}} \sup_{u \in \mathcal{U}_{|u^*}} \left[ \text{err}(\hat{f}, \mathcal{F}; u) \right], \quad (20)$$

where the infimum is again over the set of all estimators which output a function  $\hat{f} \in \mathcal{F}$  given access to distribution  $\mathcal{D}_x$  and  $k$ -comparison oracle  $\mathcal{O}_k$ . Observe that this local notion of minimax risk concerns the performance of an algorithm  $\hat{f}$  around a specific instance  $u^*$  as compared to the worst-case instance.

For any utility function  $u \in \mathcal{U}$ , we define its population maximizer  $f_u \in \text{argmax}_{f \in \mathcal{F}} U(f; u)$ . With this notation, our next theorem provides a lower bound on this local minimax risk in terms of a local modulus of continuity with respect to the set  $\mathcal{U}_{k, u^*}$ .

► **Theorem 12** (Local minimax lower bound). *For any distribution  $\mathcal{D}_x$  over feature space  $\mathcal{X}$ , utility function  $u^* \in \mathcal{U}$ , function class  $\mathcal{F}$  and order  $k$  of the comparison oracle, the local minimax risk*

$$\mathfrak{M}_k(\mathcal{F}, \mathcal{D}_x; u^*) \geq \frac{1}{2} \cdot \sup_{u_1, u_2 \in \mathcal{U}_{k, u^*}} \left( U(f_{u_1}; u_1) - U(f_{\frac{u_1+u_2}{2}}; u_1) \right). \quad (21)$$

**Proof.** Consider any two utility functions  $u_1, u_2 \in \mathcal{U}_{k, u^*}$  and let  $\bar{u} = \frac{u_1+u_2}{2}$ . We can then lower bound the minimax risk as

$$\begin{aligned} \mathfrak{M}_k(\mathcal{F}, \mathcal{D}_x; u^*) &\geq \inf_{f \in \mathcal{F}} \left( \frac{1}{2} \text{err}(f, \mathcal{F}; u_1) + \frac{1}{2} \text{err}(f, \mathcal{F}; u_2) \right) \\ &= \frac{1}{2} \text{err}(f_{\bar{u}}, \mathcal{F}; u_1) + \frac{1}{2} \text{err}(f_{\bar{u}}, \mathcal{F}; u_2) \\ &\geq \frac{1}{2} (U(f_{u_1}; u_1) - U(f_{\bar{u}}; u_1)), \end{aligned}$$

where the last equality follows by noting that  $\text{err}(f_{\bar{u}}, \mathcal{F}; u_2) \geq 0$ . Since the above holds for any choice of  $u_1, u_2$ , the desired bound follows by taking a supremum over these values. ◀

A few comments on Theorem 12 are in order. The theorem establishes that the local minimax risk  $\mathfrak{M}_k(\mathcal{F}, \mathcal{D}_x)$  is lower bounded by a local modulus of continuity,

$$\sup_{u_1, u_2 \in \mathcal{U}_{k, u^*}} \left( U(f_{u_1}; u_1) - U(f_{\frac{u_1+u_2}{2}}; u_1) \right), \quad (22)$$

which captures the worst-case variation in the performance of utility maximizers of utility in a neighborhood of  $u^*$ . For any two utilities  $u_1, u_2 \in \mathcal{U}_{k, u^*}$ , it measures the performance drop in the utility of a learner uses the maximizer  $f_{\frac{u_1+u_2}{2}}$  in place of  $f_{u_1}$  when the underlying utility is  $u_1$ .

Given this lower bound on the local minimax risk  $\mathfrak{M}_k(\mathcal{F}, \mathcal{D}_x)$ , it is natural to ask whether this local modulus of continuity exactly captures the instance-specific hardness of the problem. To this end, our next result answers this in the affirmative. In particular, it shows that for any  $u^*$ , the randomized minimax robust estimator  $p_{\text{rob}} \in \Delta_{\mathcal{F}}$ , given by

$$p_{\text{rob}} \in \text{argmin}_{p \in \Delta_{\mathcal{F}}} \sup_{u \in \mathcal{U}_{k, u^*}} \mathbb{E}_{f \sim p} [\text{err}(f, \mathcal{F}; u)], \quad (23)$$

(nearly-)obtains the same excess-risk bound as that given by the lower bound in Theorem 12.

► **Theorem 13** (Upper bounds for  $p_{\text{rob}}$ ). *For any distribution  $\mathcal{D}_x$  over feature space  $\mathcal{X}$ , utility function  $u^* \in \mathcal{U}$  and function class  $\mathcal{F}$ , the expected excess risk of the randomized estimator given by the distribution  $p_{\text{rob}} \in \Delta_{\mathcal{F}}$  is*

$$\begin{aligned} \mathbb{E}[\text{err}(p_{\text{rob}}, \mathcal{F}; u^*)] &= \sup_{p_u} (\mathbb{E}_{u' \sim p_u} [U(f_{u'}; u') - U(f_{p_u}; u')]) \\ &\leq \sup_{u_1, u_2 \in \mathcal{U}_{k, u^*}} (U(f_{u_1}; u_1) - U(f_{u_2}; u_1)), \end{aligned} \quad (24)$$

where the distribution  $p_u \in \Delta_{\mathcal{U}_{k, u^*}}$  is over the space of utility functions consistent with  $u^*$ .

We defer the proof of Theorem 13 to the full version. Compared with the lower bound of Theorem 12, the bound in (24) shows that the local minimax risk can indeed be upper bounded by a similar local modulus of continuity. Observe that while the lower bound evaluates the performance loss of the maximizer  $f_{\frac{u_1+u_2}{2}}$ , the upper bound is evaluated on  $f_{u_2}$ . While the minimax estimator  $p_{\text{rob}}$  in equation (23) is defined at the population level, we can naturally extend it to the finite sample regime as

$$\hat{p}_{\text{rob}, n} \in \underset{p \in \Delta_{\mathcal{F}}}{\text{argmin}} \sup_{u \in \hat{\mathcal{U}}_{k, u^*}} \mathbb{E}_{f \sim p} [\hat{U}(f_u; u) - \hat{U}(f; u)] \quad (25)$$

where the class of utilities  $\hat{\mathcal{U}}_{k, u^*}$  represents the set of all  $n$ -dimensional vectors in  $[0, 1]^n$  which are consistent with responses to all  $k$ -queries on the set of sampled datapoints  $S$ . Using a similar analysis as in Theorem 2, one can then upper bound the excess risk of this estimator in terms of the local modulus on the dataset  $S$  and an additional uniform convergence term.

In comparison to the Comptron procedure which uses  $O(n \log k)$  queries to the comparison oracle for estimating utilities, the estimator  $\hat{p}_{\text{rob}, n}$  uses  $O(n^k)$  queries to construct the set  $\hat{\mathcal{U}}_{k, u^*}$ . Thus, while this estimator adapts to the problem hardness, such an adaptation comes at the cost of an exponential increase in query complexity. Achieving instance-optimality by using fewer queries is an interesting question for future research.

---

## References

- 1 Pieter Abbeel and Andrew Y Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, 2004.
- 2 Sydney N Afriat. The construction of utility functions from expenditure data. *International economic review*, 8(1), 1967.
- 3 Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety. *arXiv preprint*, 2016. [arXiv:1606.06565](#).
- 4 Brenna D Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robotics and autonomous systems*, 57(5), 2009.
- 5 Maria-Florina Balcan, Andrei Broder, and Tong Zhang. Margin based active learning. In *International Conference on Computational Learning Theory*, pages 35–50. Springer, 2007.
- 6 Maria-Florina Balcan, Amit Daniely, Ruta Mehta, Ruth Uner, and Vijay V Vazirani. Learning economic parameters from revealed preferences. In *International Conference on Web and Internet Economics*, 2014.
- 7 Maria-Florina Balcan and Phil Long. Active and passive learning of linear separators under log-concave distributions. In *Conference on Learning Theory*, pages 288–316, 2013.
- 8 Peter L Bartlett and Shahrar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.
- 9 Eyal Beigman and Rakesh Vohra. Learning from revealed preference. In *Proceedings of the 7th ACM Conference on Electronic Commerce*, 2006.
- 10 Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

- 11 Tom Bylander. Learning linear threshold functions in the presence of classification noise. In *Proceedings of the seventh annual conference on Computational learning theory*, 1994.
- 12 Sanjoy Dasgupta, Adam Tauman Kalai, and Adam Tauman. Analysis of perceptron-based active learning. *Journal of Machine Learning Research*, 10(2), 2009.
- 13 Richard O Duda, Peter E Hart, and David G Stork. *Pattern classification*. John Wiley & Sons, 2012.
- 14 Johannes Fürnkranz and Eyke Hüllermeier. Preference learning and ranking by pairwise comparison. In *Preference learning*. Springer, 2010.
- 15 David Haussler. Decision theoretic generalizations of the pac model for neural net and other learning applications. *Information and computation*, 100(1), 1992.
- 16 Roni Khardon and Gabriel Wachman. Noise tolerant variants of the perceptron algorithm. *Journal of Machine Learning Research*, 8, 2007.
- 17 George A Miller. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological review*, 63(2), 1956.
- 18 Oskar Morgenstern and John Von Neumann. *Theory of games and economic behavior*. Princeton university press, 1953.
- 19 Andrew Y Ng and Stuart J Russell. Algorithms for inverse reinforcement learning. In *International Conference on Machine Learning*, volume 1, page 2, 2000.
- 20 Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6), 1958.
- 21 Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.
- 22 Shai Shalev-Shwartz, Ohad Shamir, Nathan Srebro, and Karthik Sridharan. Learnability, stability and uniform convergence. *The Journal of Machine Learning Research*, 11, 2010.
- 23 Neil Stewart, Gordon DA Brown, and Nick Chater. Absolute identification by relative judgment. *Psychological review*, 112(4), 2005.
- 24 Louis L Thurstone. The method of paired comparisons for social values. *The Journal of Abnormal and Social Psychology*, 21(4), 1927.
- 25 Songbai Yan and Chicheng Zhang. Revisiting perceptron: Efficient and label-optimal learning of halfspaces. In *Advances in Neural Information Processing Systems*, 2017.
- 26 Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, and Anind K Dey. Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, 2008.

# On Distributed Differential Privacy and Counting Distinct Elements

Lijie Chen<sup>1</sup>

Massachusetts Institute of Technology, Cambridge, MA, USA  
lijieche@mit.edu

Badih Ghazi

Google Research, Mountain View, CA, USA  
badihghazi@gmail.com

Ravi Kumar

Google Research, Mountain View, CA, USA  
ravi.k53@gmail.com

Pasin Manurangsi

Google Research, Mountain View, CA, USA  
pasin@google.com

---

## Abstract

We study the setup where each of  $n$  users holds an element from a discrete set, and the goal is to count the number of distinct elements across all users, under the constraint of  $(\epsilon, \delta)$ -differentially privacy:

- In the non-interactive *local* setting, we prove that the additive error of any protocol is  $\Omega(n)$  for any constant  $\epsilon$  and for any  $\delta$  inverse polynomial in  $n$ .
- In the *single-message shuffle* setting, we prove a lower bound of  $\tilde{\Omega}(n)$  on the error for any constant  $\epsilon$  and for some  $\delta$  inverse quasi-polynomial in  $n$ . We do so by building on the moment-matching method from the literature on distribution estimation.
- In the *multi-message shuffle* setting, we give a protocol with at most one message per user in expectation and with an error of  $\tilde{O}(\sqrt{n})$  for any constant  $\epsilon$  and for any  $\delta$  inverse polynomial in  $n$ . Our protocol is also robustly shuffle private, and our error of  $\sqrt{n}$  matches a known lower bound for such protocols.

Our proof technique relies on a new notion, that we call *dominated protocols*, and which can also be used to obtain the first non-trivial lower bounds against multi-message shuffle protocols for the well-studied problems of selection and learning parity.

Our first lower bound for estimating the number of distinct elements provides the first  $\omega(\sqrt{n})$  separation between global sensitivity and error in local differential privacy, thus answering an open question of Vadhan (2017). We also provide a simple construction that gives  $\tilde{\Omega}(n)$  separation between global sensitivity and error in *two-party* differential privacy, thereby answering an open question of McGregor et al. (2011).

**2012 ACM Subject Classification** Security and privacy → Privacy-preserving protocols

**Keywords and phrases** Differential Privacy, Shuffle Model

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.56

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/2009.09604>.

**Acknowledgements** We would like to thank Noah Golowich for numerous enlightening discussions about lower bounds in the multi-message DP<sub>shuffle</sub> model, and for helpful feedback. We also want to thank the anonymous ITCS reviewers for their helpful comments.

---

<sup>1</sup> Most of this work was done at Google Research, Mountain View, CA.



© Lijie Chen, Badih Ghazi, Ravi Kumar, and Pasin Manurangsi;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 56; pp. 56:1–56:18



Leibniz International Proceedings in Informatics

LIPIcs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

## 1 Introduction

Differential privacy (DP) [20, 19] has become a leading framework for private-data analysis, with several recent practical deployments [25, 39, 28, 3, 16, 1]. The most commonly studied DP setting is the so-called central (aka curator) model whereby a single authority (sometimes referred to as the analyst) is trusted with running an algorithm on the raw data of the users and the privacy guarantee applies to the algorithm’s output.

The absence, in many scenarios, of a clear trusted authority has motivated the study of *distributed* DP models. The most well-studied such setting is the *local* model [31] (also [44]), denoted henceforth by  $\text{DP}_{\text{local}}$ , where the privacy guarantee is enforced at each user’s output (i.e., the protocol transcript). While an advantage of the local model is its very strong privacy guarantees and minimal trust assumptions, the noise that has to be added can sometimes be quite large. This has stimulated the study of “intermediate” models that seek to achieve accuracy close to the central model while relying on more distributed trust assumptions. One such middle-ground is the so-called *shuffle* (aka *anonymous*) model [29, 8, 12, 24], where the users send messages to a *shuffler* who randomly shuffles these messages before sending them to the analyzer; the privacy guarantee is enforced on the shuffled messages (i.e., the input to the analyzer). We study both the local and the shuffle models in this work.

### 1.1 Counting Distinct Elements

A basic function in data analytics is estimating the number of distinct elements in a domain of size  $D$  held by a collection of  $n$  users, which we denote by  $\text{CountDistinct}_{n,D}$  (and simply by  $\text{CountDistinct}_n$  if there is no restriction on the universe size). Beside its use in database management systems, it is a well-studied problem in sketching, streaming, and communication complexity (e.g., [30, 9] and the references therein). In central DP, it can be easily solved with constant error using the Laplace mechanism [20]; see also [36, 15, 38, 14].

We obtain new results on  $(\epsilon, \delta)$ -DP protocols for  $\text{CountDistinct}$  in the local and shuffle settings<sup>2</sup>.

#### 1.1.1 Lower Bounds for Local DP Protocols

Our first result is a lower bound on the additive error of  $\text{DP}_{\text{local}}$  protocols<sup>3</sup> for counting distinct elements.

► **Theorem 1.** *For any  $\epsilon = O(1)$ , no public-coin  $(\epsilon, o(1/n))$ - $\text{DP}_{\text{local}}$  protocol can solve<sup>4</sup>  $\text{CountDistinct}_{n,n}$  with error  $o(n)$ .*

The lower bound in Theorem 1 is asymptotically tight<sup>5</sup>. Furthermore, it answers a question of Vadhan [42, Open Problem 9.6], who asked if there is a function with a gap of  $\omega(\sqrt{n})$  between its (global) sensitivity and the smallest achievable error by any  $\text{DP}_{\text{local}}$

<sup>2</sup> For formal definitions, please refer to Section 2. We remark that, throughout this work, we consider the *non-interactive* local model where all users apply the *same* randomizer (see Definition 15). We briefly discuss in Section 1.4 possible extensions to interactive local models. See the full version for how to generalize our results to the relaxed setting where each user can apply different randomizers to their inputs.

<sup>3</sup> See Section 2 for the formal (standard) definition of public-coin DP protocols. Note that private-coin protocols are a sub-class of public-coin protocols, so all of our lower bounds apply to private-coin protocols as well.

<sup>4</sup> Throughout this work, we say that a randomized algorithm solves a problem with error  $e$  if with probability 0.99 it incurs error at most  $e$ .

<sup>5</sup> The trivial algorithm that always outputs 0 incurs an error  $n$ .

protocol.<sup>6</sup> As the global sensitivity of the number of distinct elements is 1, Theorem 1 exhibits a (natural) function for which this gap is as large as  $\Omega(n)$ . While Theorem 1 applies to the constant  $\varepsilon$  regime, it turns out we can prove a lower bound for much less private protocols (i.e., having a much larger  $\varepsilon$  value) at the cost of polylogarithmic factors in the error:

► **Theorem 2.** *For some  $\varepsilon = \ln(n) - O(\ln \ln n)$  and  $D = \Theta(n / \text{polylog}(n))$ , no public-coin  $(\varepsilon, n^{-\omega(1)})$ -DP<sub>local</sub> protocol can solve  $\text{CountDistinct}_{n,D}$  with error  $o(D)$ .*

To prove Theorem 2, we build on the *moment matching* method from the literature on (non-private) distribution estimation, namely [43, 45], and tailor it to  $\text{CountDistinct}$  in the DP<sub>local</sub> setting (see Section 3.1 for more details on this connection). The bound on the privacy parameter  $\varepsilon$  in Theorem 2 turns out to be very close to tight: the error drops quadratically when  $\varepsilon$  exceeds  $\ln n$ . This is shown in the next theorem:

► **Theorem 3.** *There is a  $(\ln(n) + O(1))$ -DP<sub>local</sub> protocol solving  $\text{CountDistinct}_{n,n}$  with error  $O(\sqrt{n})$ .*

### 1.1.2 Lower Bounds for Single-Message Shuffle DP Protocols

In light of the negative result in Theorem 2, a natural question is whether  $\text{CountDistinct}$  can be solved in a weaker distributed DP setting such as the shuffle model. It turns out that this is not possible using any shuffle protocol where each user sends no more than 1 message (for brevity, we will henceforth denote this class by  $\text{DP}_{\text{shuffle}}^1$ , and more generally denote by  $\text{DP}_{\text{shuffle}}^k$  the variant where each user can send up to  $k$  messages). Note that the class  $\text{DP}_{\text{shuffle}}^1$  includes any method obtained by taking a DP<sub>local</sub> protocol and applying the so-called *amplification by shuffling* results of [24, 6].

In the case where  $\varepsilon$  is any constant and  $\delta$  is inverse quasi-polynomial in  $n$ , the improvement in the error for  $\text{DP}_{\text{shuffle}}^1$  protocols compared to DP<sub>local</sub> is at most polylogarithmic factors:

► **Theorem 4.** *For all  $\varepsilon = O(1)$ , there are  $\delta = 2^{-\text{polylog}(n)}$  and  $D = n / \text{polylog}(n)$  such that no public-coin  $(\varepsilon, \delta)$ -DP<sub>shuffle</sub><sup>1</sup> protocol can solve  $\text{CountDistinct}_{n,D}$  with error  $o(D)$ .*

We note that Theorem 4 essentially answers a more general variant of Vadhan’s question: it shows that even for  $\text{DP}_{\text{shuffle}}^1$  protocols (which include DP<sub>local</sub> protocols as a sub-class) the gap between sensitivity and the error can be as large as  $\tilde{\Omega}(n)$ .

The proof of Theorem 4 follows by combining Theorem 2 with the following connection between DP<sub>local</sub> and  $\text{DP}_{\text{shuffle}}^1$ :

► **Lemma 5.** *For any  $\varepsilon = O(1)$  and  $\delta \leq \delta_0 \leq 1/n$ , if the randomizer  $R$  is  $(\varepsilon, \delta)$ -DP<sub>shuffle</sub><sup>1</sup> on  $n$  users, then  $R$  is  $(\ln n - \ln(\Theta_\varepsilon(\log \delta_0^{-1} / \log \delta^{-1})), \delta_0)$ -DP<sub>local</sub>.*

We remark that Lemma 5 provides a stronger quantitative bound than the qualitatively similar connections in [12, 27]; specifically, we obtain the term  $\ln(\Theta_\varepsilon(\log \delta_0^{-1} / \log \delta^{-1}))$ , which was not present in the aforementioned works. This turns out to be crucial for our purposes, as this term gives the  $O(\ln \ln n)$  term necessary to apply Theorem 2.

<sup>6</sup> To the best of our knowledge, the largest previously known gap between global sensitivity and error was  $O(\sqrt{n})$ , which is achieved, e.g., by binary summation [11]. For  $\text{CountDistinct}$ , the lower bound of [21] on pan-private algorithms against two intrusions along with the equivalence shown by [2] between this model and sequential local DP, imply a lower bound of  $\Omega(n)$  against *pure* DP protocols. A lower bound against approximate DP protocols can then be obtained via the transformation of [10]; however, this lower bound would only hold for an  $\varepsilon$  bounded strictly below one (e.g.,  $1/4$ ), whereas our lower bound in Theorem 1 holds for  $\varepsilon$  an arbitrarily large constant.

### 1.1.3 A Communication-Efficient Shuffle DP Protocol

In contrast with Theorem 4, Balcer et al. [5] recently gave a  $\text{DP}_{\text{shuffle}}$  protocol for  $\text{CountDistinct}_{n,D}$  with error  $O(\sqrt{D})$ . Their protocol sends  $\Omega(D)$  messages per user. We instead show that an error of  $\tilde{O}(\sqrt{D})$  can still be guaranteed with each user sending *in expectation* at most one message each of length  $O(\log D)$  bits.

► **Theorem 6.** *For all  $\varepsilon \leq O(1)$  and  $\delta \leq 1/n$ , there is a public-coin  $(\varepsilon, \delta)$ - $\text{DP}_{\text{shuffle}}$  protocol that solves  $\text{CountDistinct}_n$  with error  $\sqrt{\min(n, D)} \cdot \text{poly}(\log(1/\delta)/\varepsilon)$  where the expected number of messages sent by each user is at most one.*

In the special case where  $D = o(n / \text{poly}(\varepsilon^{-1} \log(\delta^{-1})))$ , we moreover obtain a *private-coin*  $\text{DP}_{\text{shuffle}}$  protocol achieving the same guarantees as in Theorem 6 (see the full version for a formal statement). Note that Theorem 6 is in sharp contrast with the lower bound shown in Theorem 4 for  $\text{DP}_{\text{shuffle}}^1$  protocols. Indeed, for  $\delta$  inverse quasi-polynomial in  $n$ , the former implies a public-coin protocol with less than one message per-user *in expectation* having error  $\tilde{O}(\sqrt{n})$  whereas the latter proves that no such protocol exists, even with error as large as  $\tilde{\Omega}(n)$ , if we restrict each user to send one message *in the worst case*.

A strengthening of  $\text{DP}_{\text{shuffle}}$  protocols called *robust  $\text{DP}_{\text{shuffle}}$  protocols*<sup>7</sup> was studied by [5], who proved an  $\Omega(\sqrt{\min(D, n)})$  lower bound on the error of any protocol solving  $\text{CountDistinct}_{n,D}$ . Our protocols are robust  $\text{DP}_{\text{shuffle}}$  and, therefore, achieve the optimal error (up to polylogarithmic factors) among all robust  $\text{DP}_{\text{shuffle}}$  protocols, while only sending at most one message per user in expectation.

## 1.2 Dominated Protocols and Multi-Message Shuffle DP Protocols

The technique underlying the proof of Theorem 1 can be extended beyond  $\text{DP}_{\text{local}}$  protocols for  $\text{CountDistinct}$ . It applies to a broader category of protocols that we call *dominated*, defined as follows:

► **Definition 7.** *We say that a randomizer  $R: \mathcal{X} \rightarrow \mathcal{M}$  is  $(\varepsilon, \delta)$ -dominated, if there exists a distribution  $\mathcal{D}$  on  $\mathcal{M}$  such that for all  $x \in \mathcal{X}$  and all  $E \subseteq \mathcal{M}$ ,*

$$\Pr[R(x) \in E] \leq e^\varepsilon \cdot \Pr_{\mathcal{D}}[E] + \delta.$$

*In this case, we also say  $R$  is  $(\varepsilon, \delta)$ -dominated by  $\mathcal{D}$ . We define  $(\varepsilon, \delta)$ -dominated protocols in the same way as  $(\varepsilon, \delta)$ - $\text{DP}_{\text{local}}$ , except that we require the randomizer to be  $(\varepsilon, \delta)$ -dominated instead of being  $(\varepsilon, \delta)$ -DP.*

Note that an  $(\varepsilon, \delta)$ - $\text{DP}_{\text{local}}$  randomizer  $R$  is  $(\varepsilon, \delta)$ -dominated: we can fix a  $y^* \in \mathcal{X}$  and take  $\mathcal{D} = R(y^*)$ . Therefore, our new definition is a relaxation of  $\text{DP}_{\text{local}}$ .

We show that *multi-message*  $\text{DP}_{\text{shuffle}}$  protocols are dominated, which allows us to prove the first non-trivial lower bounds against  $\text{DP}_{\text{shuffle}}^{O(1)}$  protocols.

Before formally stating this connection, we recall why known lower bounds against  $\text{DP}_{\text{shuffle}}^1$  protocols [12, 27, 4] do not extend to  $\text{DP}_{\text{shuffle}}^{O(1)}$  protocols.<sup>8</sup> These prior works use the connection stating that any  $(\varepsilon, \delta)$ - $\text{DP}_{\text{shuffle}}^1$  protocol is also  $(\varepsilon + \ln n, \delta)$ - $\text{DP}_{\text{local}}$  [12,

<sup>7</sup> Roughly speaking, they are  $\text{DP}_{\text{shuffle}}$  protocols whose transcript remains private even if a constant fraction of users drop out from the protocol.

<sup>8</sup> We remark that [26] developed a technique for proving lower bounds on the *communication complexity* (i.e., the number of bits sent per user) for multi-message protocols. Their techniques do not apply to our setting as our lower bounds are in terms of the number of messages, and do not put any restriction on the message length. Furthermore, their technique only applies to *pure*-DP where  $\delta = 0$ , whereas ours applies also to *approximate*-DP where  $\delta > 0$ .



Theorem 6.2]. It thus suffices for them to prove lower bounds for  $\text{DP}_{\text{local}}$  protocols with low privacy requirement (i.e.,  $(\varepsilon + \ln n, \delta)\text{-DP}_{\text{local}}$ ), for which lower bound techniques are known or developed. For  $\varepsilon\text{-DP}_{\text{shuffle}}^1$  protocols, [4] showed that they are also  $\varepsilon\text{-DP}_{\text{local}}$ ; therefore, lower bounds on  $\text{DP}_{\text{local}}$  protocols automatically translate to lower bounds on pure- $\text{DP}_{\text{shuffle}}^1$  protocols. To apply this proof framework to  $\text{DP}_{\text{shuffle}}^{O(1)}$  protocols, a natural first step would be to connect  $\text{DP}_{\text{shuffle}}^{O(1)}$  protocols to  $\text{DP}_{\text{local}}$  protocols. However, as observed by [4, Section 4.1], there exists an  $\varepsilon\text{-DP}_{\text{shuffle}}^{O(1)}$  protocol that is not  $\text{DP}_{\text{local}}$  for any privacy parameter. That is, there is no analogous connection between  $\text{DP}_{\text{local}}$  protocols and multi-message  $\text{DP}_{\text{shuffle}}$  protocols, even if the latter can only send  $O(1)$  messages per user.

In contrast, the next lemma captures the connection between multi-message  $\text{DP}_{\text{shuffle}}$  and dominated protocols.

► **Lemma 8.** *If  $R$  is  $(\varepsilon, \delta)\text{-DP}_{\text{shuffle}}^k$  on  $n$  users, then it is  $(\varepsilon + k(1 + \ln n), \delta)\text{-dominated}$ .*

By considering dominated protocols and using Lemma 8, we obtain the first lower bounds for *multi-message*  $\text{DP}_{\text{shuffle}}$  protocols for two well-studied problems: **Selection** and **ParityLearning**.

### 1.2.1 Lower Bounds for Selection

The **Selection** problem on  $n$  users is defined as follows. The  $i$ th user has an input  $x_i \in \{0, 1\}^D$  and the goal is to output an index  $j \in [D]$  such that  $\sum_{i=1}^n x_{i,j} \geq \left( \max_{j^*} \sum_{i=1}^n x_{i,j^*} \right) - n/10$ .

**Selection** is well-studied in DP (e.g., [17, 40, 41]) and its variants are useful primitives for several statistical and algorithmic problems including feature selection, hypothesis testing and clustering. In central DP, the exponential mechanism of [35] yields an  $\varepsilon\text{-DP}$  algorithm for **Selection** when  $n = O_\varepsilon(\log D)$ . On the other hand, it is known that any  $(\varepsilon, \delta)\text{-DP}_{\text{local}}$  protocol for **Selection** with  $\varepsilon = O(1)$  and  $\delta = O(1/n^{1.01})$  requires  $n = \Omega(D \log D)$  users [41]. Moreover, [12] obtained a  $(\varepsilon, 1/n^{O(1)})\text{-DP}_{\text{shuffle}}^D$  protocol for  $n = \tilde{O}_\varepsilon(\sqrt{D})$ . By contrast, for  $\text{DP}_{\text{shuffle}}^1$  protocols, a lower bound of  $\Omega(D^{1/17})$  was obtained in [12] and improved to  $\Omega(D)$  in [27].

The next theorem give a lower bounds for **Selection** that holds against approximate- $\text{DP}_{\text{shuffle}}^k$  protocols. To the best of our knowledge, this is the first lower bound even for  $k = 2$  (and even for the special case of pure protocols, where  $\delta = 0$ ).

► **Theorem 9.** *For any  $\varepsilon = O(1)$ , any public-coin  $(\varepsilon, o(1/D))\text{-DP}_{\text{shuffle}}^k$  protocol that solves **Selection** requires  $n \geq \Omega\left(\frac{D}{k}\right)$ .*

We remark that combining the advanced composition theorem for DP and known  $\text{DP}_{\text{shuffle}}$  aggregation algorithms, one can obtain a  $(\varepsilon, 1/\text{poly}(n))\text{-DP}_{\text{shuffle}}^k$  protocol for **Selection** with  $\tilde{O}(D/\sqrt{k})$  samples for any  $k \leq D$  (see the full version for details).

### 1.2.2 Lower Bounds for Parity Learning

In **ParityLearning**, there is a hidden random vector  $s \in \{0, 1\}^D$ , each user gets a random vector  $x \in \{0, 1\}^D$  together with the inner product  $\langle s, x \rangle$  over  $\mathbb{F}_2$ , and the goal is to recover  $s$ . This problem is well-known for separating PAC learning from the Statistical Query (SQ) learning model [32]. In DP, it was studied by [31] who gave a central DP protocol (also based on the exponential mechanism) computing it for  $n = O(D)$ , and moreover proved a lower bound of  $n = 2^{\Omega(D)}$  for any  $\text{DP}_{\text{local}}$  protocol, thus obtaining the first exponential separation between the central and local settings.

We give a lower bound for ParityLearning that hold against approximate- $\text{DP}_{\text{shuffle}}^k$  protocols:

► **Theorem 10.** *For any  $\varepsilon = O(1)$ , if  $P$  is a public-coin  $(\varepsilon, o(1/n))$ - $\text{DP}_{\text{shuffle}}^k$  protocol that solves ParityLearning with probability at least 0.99, then  $n \geq \Omega(2^{D/(k+1)})$ .*

Our lower bounds for ParityLearning can be generalized to the Statistical Query (SQ) learning framework of [32] (see the full version for more details).

### Independent Work

In a recent concurrent work, Cheu and Ullman [13] proved that robust  $\text{DP}_{\text{shuffle}}$  protocols solving Selection and ParityLearning require  $\Omega(\sqrt{D})$  and  $\Omega(2^{\sqrt{D}})$  samples, respectively. Their results have no restriction on the number of messages sent by each user, but they only hold against the special case of *robust* protocols. Our results provide stronger lower bounds when the number of messages per user is less than  $\sqrt{D}$ , and apply to the most general  $\text{DP}_{\text{shuffle}}$  model without the robustness restriction.

### 1.3 Lower Bounds for Two-Party DP Protocols

Finally, we consider another model of distributed DP, called the *two-party* model [33], denoted  $\text{DP}_{\text{two-party}}$ . In this model, there are two parties, each holding part of the dataset. The DP guarantee is enforced on the view of each party (i.e., the transcript, its private randomness, and its input). See the full version for a formal treatment.

McGregor et al. [33] studied the  $\text{DP}_{\text{two-party}}$  and proved an interesting separation of  $\Omega_\varepsilon(n)$  between the global sensitivity and  $\varepsilon$ -DP protocol in this model. However, this lower bound does not extend to the approximate-DP case (where  $\delta > 0$ ); in this case, the largest known gap (also proved in [33]) is only  $\tilde{\Omega}_\varepsilon(\sqrt{n})$ , and it was left as an open question if this can be improved<sup>9</sup>. We answer this question by showing that the gap of  $\tilde{\Omega}_\varepsilon(n)$  holds even against approximate-DP protocols:

► **Theorem 11.** *For any  $\varepsilon = O(1)$  and any sufficiently large  $n \in \mathbb{N}$ , there is a function  $f: \{0, 1\}^{2n} \rightarrow \mathbb{R}$  whose global sensitivity is one and such that no  $(\varepsilon, o(1/n))$ - $\text{DP}_{\text{two-party}}$  protocol can compute  $f$  to within an error of  $o(n/\log n)$ .*

The above bound is tight up to a logarithmic factors in  $n$ , as it is trivial to achieve an error of  $n$ .

The proof of Theorem 11 is unlike others in the paper; in fact, we only employ simple reductions starting from the hardness of inner product function already shown in [33]. Specifically, our function is a sum of blocks of inner product modulo 2. While this function is not symmetric, we show that it can be easily symmetrized (see the full version for details).

### 1.4 Discussions and Open Questions

In this work, we study DP in distributed models, including the local and shuffle settings. By building on the moment matching method and using the newly defined notion of dominated protocols, we give novel lower bounds in both models for three fundamental problems: CountDistinct, Selection, and ParityLearning. While our lower bounds are (nearly) tight in a large setting of parameters, there are still many interesting open questions, three of which we highlight below:

<sup>9</sup> The conference version of the paper [33] actually claimed to also have a lower bound  $\Omega_\varepsilon(n)$  for the approximate-DP case as well. However, it was later found to be incorrect; see [34] for more discussions.

- **DP<sub>shuffle</sub> Lower Bounds for Protocols with Unbounded Number of Messages.** Our connection between DP<sub>shuffle</sub> and dominated protocols becomes weaker as  $k \rightarrow \infty$  (Lemma 8). As a result, it cannot be used to establish lower bounds against DP<sub>shuffle</sub> protocols with a possibly unbounded number of messages. In fact, we are not aware of any separation between central DP and DP<sub>shuffle</sub> without a restriction on the number of messages and without the robustness restriction. This remains a fundamental open question. (In contrast, separations between central DP and DP<sub>local</sub> are well-known, even for basic functions such as binary summation [11].)
- **Lower Bounds against Interactive Local/Shuffle Model.** Our lower bounds hold in the *non-interactive* local and shuffle DP models, where all users send their messages simultaneously in a single round. While it seems plausible that our lower bounds can be extended to the *sequentially interactive* local DP model [17] (where each user speaks once but not simultaneously), it is unclear how to extend them to the fully interactive local DP model.  
The situation for DP<sub>shuffle</sub> however is more complicated. We remark that certain definitions could lead to the model being as powerful as the central model (in terms of achievable accuracy and putting aside communication constraints); see e.g., [29] on how to perform secure computations under a certain definition of the shuffle model. A very recent work provides a formal treatment of an interactive setting for the shuffle model [7].
- **DP<sub>shuffle</sub><sup>1</sup> Lower Bounds for CountDistinct with Larger  $\delta$ .** All but one of our lower bounds hold as long as  $\delta = n^{-\omega(1)}$ , which is a standard assumption in the DP literature. The only exception is that of Theorem 4, which requires  $\delta = 2^{-\Omega(\log^c n)}$  for some constant  $c > 0$ . It is interesting whether this can be relaxed to  $\delta = n^{-\omega(1)}$ .

## 2 Preliminaries

### 2.1 Notation

For a function  $f: \mathcal{X} \rightarrow \mathbb{R}$ , a distribution  $\mathcal{D}$  on  $\mathcal{X}$ , and an element  $z \in \mathcal{X}$ , we use  $f(\mathcal{D})$  to denote  $\mathbb{E}_{x \leftarrow \mathcal{D}} [f(x)]$  and  $\mathcal{D}_z$  to denote  $\Pr_{x \leftarrow \mathcal{D}} [x = z]$ . For a subset  $E \subseteq \mathcal{X}$ , we use  $\mathcal{D}_E$  to denote  $\sum_{z \in E} \mathcal{D}_z = \Pr_{x \leftarrow \mathcal{D}} [x \in E]$ . We also use  $\mathcal{U}_D$  to denote the uniform distribution over  $\{0, 1\}^D$ .

For two distributions  $\mathcal{D}_1$  and  $\mathcal{D}_2$  on sets  $\mathcal{X}$  and  $\mathcal{Y}$  respectively, we use  $\mathcal{D}_1 \otimes \mathcal{D}_2$  to denote their product distribution over  $\mathcal{X} \times \mathcal{Y}$ . For two random variables  $X$  and  $Y$  supported on  $\mathbb{R}^D$  for  $D \in \mathbb{N}$ , we use  $X + Y$  to denote the random variable distributed as a sum of two independent samples from  $X$  and  $Y$ . For any set  $\mathcal{S}$ , we denote by  $\mathcal{S}^*$  the set consisting of sequences on  $\mathcal{S}$ , i.e.,  $\mathcal{S}^* = \cup_{n \geq 0} \mathcal{S}^n$ . For  $x \in \mathbb{R}$ , let  $[x]_+$  denote  $\max(x, 0)$ . For a predicate  $P$ , we use  $\mathbb{K}[P]$  to denote the corresponding Boolean value of  $P$ , that is,  $\mathbb{K}[P] = 1$  if  $P$  is true, and 0 otherwise.

For a distribution  $\mathcal{D}$  on a finite set  $\mathcal{X}$  and an event  $\mathcal{E} \subseteq \mathcal{X}$  such that  $\Pr_{z \leftarrow \mathcal{D}} [z \in \mathcal{E}] > 0$ , we use  $\mathcal{D}|\mathcal{E}$  to denote the conditional distribution such that

$$(\mathcal{D}|\mathcal{E})_z = \begin{cases} \frac{\mathcal{D}_z}{\Pr_{z \leftarrow \mathcal{D}} [z \in \mathcal{E}]} & \text{if } z \in \mathcal{E}, \\ 0 & \text{otherwise.} \end{cases}$$

Slightly overloading the notation, we also use  $\alpha \cdot \mathcal{D}_1 + (1 - \alpha) \cdot \mathcal{D}_2$  to denote the mixture of distributions  $\mathcal{D}_1$  and  $\mathcal{D}_2$  with mixing weights  $\alpha$  and  $(1 - \alpha)$  respectively. Whether  $+$  means mixture or convolution will be clear from the context unless explicitly stated.

## 2.2 Differential Privacy

We now recall the basics of differential privacy that we will need. Fix a finite set  $\mathcal{X}$ , the space of user reports. A *dataset*  $X$  is an element of  $\mathcal{X}^*$ , namely a tuple consisting of elements of  $\mathcal{X}$ . Let  $\text{hist}(X) \in \mathbb{N}^{|\mathcal{X}|}$  be the *histogram* of  $X$ : for any  $x \in \mathcal{X}$ , the  $x$ th component of  $\text{hist}(X)$  is the number of occurrences of  $x$  in the dataset  $X$ . We will consider datasets  $X, X'$  to be *equivalent* if they have the same histogram (i.e., the ordering of the elements  $x_1, \dots, x_n$  does not matter). For a multiset  $\mathcal{S}$  whose elements are in  $\mathcal{X}$ , we will also write  $\text{hist}(\mathcal{S})$  to denote the histogram of  $\mathcal{S}$  (so that the  $x$ th component is the number of copies of  $x$  in  $\mathcal{S}$ ).

Let  $n \in \mathbb{N}$ , and consider a dataset  $X = (x_1, \dots, x_n) \in \mathcal{X}^n$ . For an element  $x \in \mathcal{X}$ , let  $f_X(x) = \frac{\text{hist}(X)_x}{n}$  be the *frequency* of  $x$  in  $X$ , namely the fraction of elements of  $X$  that are equal to  $x$ . Two datasets  $X, X'$  are said to be *neighboring* if they differ in a single element, meaning that we can write (up to equivalence)  $X = (x_1, x_2, \dots, x_n)$  and  $X' = (x'_1, x_2, \dots, x_n)$ . In this case, we write  $X \sim X'$ . Let  $\mathcal{Z}$  be a set; we now define the differential privacy of a randomized function  $P: \mathcal{X}^n \rightarrow \mathcal{Z}$  as follows.

► **Definition 12** (Differential privacy (DP) [20, 19]). *A randomized algorithm  $P: \mathcal{X}^n \rightarrow \mathcal{Z}$  is  $(\varepsilon, \delta)$ -DP if for every pair of neighboring datasets  $X \sim X'$  and for every set  $\mathcal{S} \subseteq \mathcal{Z}$ , we have*

$$\Pr[P(X) \in \mathcal{S}] \leq e^\varepsilon \cdot \Pr[P(X') \in \mathcal{S}] + \delta,$$

where the probabilities are taken over the randomness in  $P$ . Here,  $\varepsilon \geq 0$  and  $\delta \in [0, 1]$ .

If  $\delta = 0$ , then we use  $\varepsilon$ -DP for brevity and informally refer to it as *pure-DP*; if  $\delta > 0$ , we refer to it as *approximate-DP*. We will use the following post-processing property of DP.

► **Lemma 13** (Post-processing, e.g., [22]). *If  $P$  is  $(\varepsilon, \delta)$ -DP, then for every randomized function  $A$ , the composed function  $A \circ P$  is  $(\varepsilon, \delta)$ -DP.*

## 2.3 Shuffle Model

We briefly review the *shuffle model* of DP [8, 24, 12]. The input to the model is a dataset  $(x_1, \dots, x_n) \in \mathcal{X}^n$ , where item  $x_i \in \mathcal{X}$  is held by user  $i$ . A protocol  $P: \mathcal{X} \rightarrow \mathcal{Z}$  in the shuffle model consists of three algorithms:

- The *local randomizer*  $R: \mathcal{X} \rightarrow \mathcal{M}^*$  takes as input the data of one user,  $x_i \in \mathcal{X}$ , and outputs a sequence  $(y_{i,1}, \dots, y_{i,m_i})$  of *messages*; here  $m_i$  is a positive integer.  
To ease discussions in the paper, we will further assume that the randomizer  $R$  pre-shuffles its messages. That is, it applies a random permutation  $\pi: [m_i] \rightarrow [m_i]$  to the sequence  $(y_{i,1}, \dots, y_{i,m_i})$  before outputting it.<sup>10</sup>
- The *shuffler*  $S: \mathcal{M}^* \rightarrow \mathcal{M}^*$  takes as input a sequence of elements of  $\mathcal{M}$ , say  $(y_1, \dots, y_m)$ , and outputs a random permutation, i.e., the sequence  $(y_{\pi(1)}, \dots, y_{\pi(m)})$ , where  $\pi \in S_m$  is a uniformly random permutation on  $[m]$ . The input to the shuffler will be the concatenation of the outputs of the local randomizers.
- The *analyzer*  $A: \mathcal{M}^* \rightarrow \mathcal{Z}$  takes as input a sequence of elements of  $\mathcal{M}$  (which will be taken to be the output of the shuffler) and outputs an answer in  $\mathcal{Z}$  that is taken to be the output of the protocol  $P$ .

<sup>10</sup>Therefore, for every  $x \in \mathcal{X}$  and any two tuples  $z_1, z_2 \in \mathcal{M}^*$  that are equivalent up to a permutation,  $R(x)$  outputs them with the same probability.

We will write  $P = (R, S, A)$  to denote the protocol whose components are given by  $R$ ,  $S$ , and  $A$ . The main distinction between the shuffle and local model is the introduction of the shuffler  $S$  between the local randomizer and the analyzer. As in the local model, the analyzer is untrusted in the shuffle model; hence privacy must be guaranteed with respect to the input to the analyzer, i.e., the output of the shuffler. Formally, we have:

► **Definition 14** (DP in the Shuffle Model, [24, 12]). *A protocol  $P = (R, S, A)$  is  $(\varepsilon, \delta)$ -DP if, for any dataset  $X = (x_1, \dots, x_n)$ , the algorithm*

$$(x_1, \dots, x_n) \mapsto S(R(x_1), \dots, R(x_n))$$

*is  $(\varepsilon, \delta)$ -DP.*

Notice that the output of  $S(R(x_1), \dots, R(x_n))$  can be simulated by an algorithm that takes as input the *multiset* consisting of the union of the elements of  $R(x_1), \dots, R(x_n)$  (which we denote as  $\bigcup_i R(x_i)$ , with a slight abuse of notation) and outputs a uniformly random permutation of them. Thus, by Lemma 13, it can be assumed without loss of generality for privacy analyses that the shuffler simply outputs the multiset  $\bigcup_i R(x_i)$ . For the purpose of analyzing the accuracy of the protocol  $P = (R, S, A)$ , we define its *output* on the dataset  $X = (x_1, \dots, x_n)$  to be  $P(X) := A(S(R(x_1), \dots, R(x_n)))$ . We also remark that the case of *local DP*, formalized in Definition 15, is a special case of the shuffle model where the shuffler  $S$  is replaced by the identity function:

► **Definition 15** (Local DP [31]). *A protocol  $P = (R, A)$  is  $(\varepsilon, \delta)$ -DP in the local model (or  $(\varepsilon, \delta)$ -locally DP) if the function  $x \mapsto R(x)$  is  $(\varepsilon, \delta)$ -DP.*

We say that the *output* of the protocol  $P$  on an input dataset  $X = (x_1, \dots, x_n)$  is  $P(X) := A(R(x_1), \dots, R(x_n))$ .

We denote DP in the shuffle model by  $\text{DP}_{\text{shuffle}}$ , and the special case where each user can send at most<sup>11</sup>  $k$  messages by  $\text{DP}_{\text{shuffle}}^k$ . We denote DP in the local model by  $\text{DP}_{\text{local}}$ .

## Public-Coin DP

The default setting for local and shuffle models is private-coin, i.e., there is no randomness shared between the randomizers and the analyzer. We will also study the public-coin variants of the local and shuffle models. In the public-coin setting, each local randomizer also takes a public random string  $\alpha \leftarrow \{0, 1\}^*$  as input. The analyzer is also given the public random string  $\alpha$ . We use  $R_\alpha(x)$  to denote the local randomizer with public random string being fixed to  $\alpha$ . At the start of the protocol, all users jointly sample a public random string from a publicly known distribution  $\mathcal{D}_{\text{pub}}$ .

Now, we say that a protocol  $P = (R, A)$  is  $(\varepsilon, \delta)$ -DP in the *public-coin local model*, if the function

$$x \xrightarrow{\alpha \leftarrow \mathcal{D}_{\text{pub}}} (\alpha, R_\alpha(x))$$

is  $(\varepsilon, \delta)$ -DP.

<sup>11</sup> We may assume w.l.o.g. that each user sends *exactly*  $k$  messages; otherwise, we may define a new symbol  $\perp$  and make each user send  $\perp$  messages so that the number of messages becomes exactly  $k$ .

Similarly, we say that a protocol  $P = (R, S, A)$  is  $(\varepsilon, \delta)$ -DP in the public-coin shuffle model, if for any dataset  $X = (x_1, \dots, x_n)$ , the algorithm

$$(x_1, \dots, x_n) \xrightarrow{\alpha \leftarrow \mathcal{D}_{\text{pub}}} (\alpha, S(R_\alpha(x_1), \dots, R_\alpha(x_n)))$$

is  $(\varepsilon, \delta)$ -DP.

## 2.4 Useful Divergences

We will make use of two important divergences between distributions, the KL-divergence and the  $\chi^2$ -divergence, defined as

$$KL(P||Q) = \mathbb{E}_{z \leftarrow P} \log \left( \frac{P_z}{Q_z} \right) \quad \text{and} \quad \chi^2(P||Q) = \mathbb{E}_{z \leftarrow Q} \left[ \frac{P_z - Q_z}{Q_z} \right]^2.$$

We will also use Pinsker's inequality, whereby the total variation distance lower-bounds the KL-divergence:

$$KL(P||Q) \geq \frac{2}{\ln 2} \|P - Q\|_{TV}^2.$$

## 2.5 Fourier Analysis

We now review some basic Fourier analysis and then introduce two inequalities that will be heavily used in our proofs. For a function  $f: \{0, 1\}^D \rightarrow \mathbb{R}$ , its Fourier transform is given by the function  $\hat{f}(S) := \mathbb{E}_{x \leftarrow \mathcal{U}_D} [f(x) \cdot (-1)^{\sum_{i \in S} x_i}]$ . We also define  $\|f\|_2^2 = \mathbb{E}_{x \leftarrow \mathcal{U}_D} [f(x)^2]$ . For  $k \in \mathbb{N}$ ,

we define the *level- $k$  Fourier weight* as  $\mathbf{W}^k[f] := \sum_{S \subseteq [D], |S|=k} \hat{f}(S)^2$ . For convenience, for

$s \in \{0, 1\}^D$ , we will also write  $\hat{f}(s)$  to denote  $f(\chi_s)$ , where  $\chi_s$  is the set  $\{i: i \in [D] \wedge s_i = 1\}$ . One key technical lemma is the Level-1 Inequality from [37], which was also used in [27].

► **Lemma 16** (Level-1 Inequality). *Suppose  $f: \{0, 1\}^D \rightarrow \mathbb{R}_{\geq 0}$  is a non-negative-valued function with  $f(x) \in [0, L]$  for all  $x \in \{0, 1\}^D$ , and  $\mathbb{E}_{x \sim \mathcal{U}_D} [f(x)] \leq 1$ . Then,  $\mathbf{W}^1[f] \leq 6 \ln(L + 1)$ .*

We also need the standard Parseval's identity.

► **Lemma 17** (Parseval's Identity). *For all functions  $f: \{0, 1\}^D \rightarrow \mathbb{R}$ ,*

$$\|f\|_2^2 = \sum_{S \subseteq [D]} \hat{f}(S)^2.$$

## 3 Overview of Techniques

In this section, we describe the main intuition behind our lower bounds. As alluded to in Section 1, we give two different proofs of the lower bounds for **CountDistinct** in the  $\text{DP}_{\text{local}}$  and  $\text{DP}_{\text{shuffle}}$  settings, each with its own advantages:

- **Proof via Moment Matching.** Our first proof is technically the hardest in our work. It applies to the much more challenging low-privacy setting (i.e.,  $(\ln n - O(\ln \ln n), \delta)$ - $\text{DP}_{\text{local}}$ ), and shows an  $\Omega(n / \text{polylog}(n))$  lower bound on the additive error (Theorem 2). Together with our new improved connection between  $\text{DP}_{\text{shuffle}}^1$  and  $\text{DP}_{\text{local}}$  (Lemma 5), it also implies the same lower bound for protocols in the  $\text{DP}_{\text{shuffle}}^1$  model. The key ideas behind the first proof will be discussed in Section 3.1.

- **Proof via Dominated Protocols.** Our second proof has the advantage of giving the optimal  $\Omega(n)$  lower bound on the additive error (Theorem 1), but only in the constant privacy regime (i.e.,  $(O(1), \delta)$ -DP<sub>local</sub>), and it is relatively simple compared to the first proof.

Moreover, the second proof technique is very general and is a conceptual contribution: it can be applied to show lower bounds for other fundamental problems (i.e., **Selection** and **ParityLearning**; Theorems 9 and 10) against multi-message DP<sub>shuffle</sub> protocols. We will highlight the intuition behind the second proof in Section 3.2.

While our lower bounds also work for the public-coin DP<sub>shuffle</sub> models, throughout this section, we focus on private-coin models in order to simplify the presentation. The full proofs extending to public-coin protocols are given in the full version.

### 3.1 Lower Bounds for CountDistinct via Moment Matching

To clearly illustrate the key ideas behind the first proof, we will focus on the pure-DP case where each user can only send  $O(\log n)$  bits. In the full version, we generalize the proof to approximate-DP and remove the restriction on communication complexity.

► **Theorem 18** (A Weaker Version of Theorem 2). *For  $\varepsilon = \ln(n/\log^7 n)$  and  $D = n/\log^5 n$ , no  $\varepsilon$ -DP<sub>local</sub> protocol where each user sends  $O(\log n)$  bits can solve **CountDistinct** <sub>$n, D$</sub>  with error  $o(D)$ .*

Throughout our discussion, we use  $R : [D] \rightarrow \mathcal{M}$  to denote a  $\ln(n/\log^7 n)$ -DP<sub>local</sub> randomizer. By the communication complexity condition of Theorem 18, we have that  $|\mathcal{M}| \leq \text{poly}(n)$ .

Our proof is inspired by the lower bounds for estimating distinct elements in the property testing model, e.g., [43, 45]. In particular, we use the so-called *Poissonization* trick. To discuss this trick, we start with some notation. For a vector  $\vec{\lambda} \in \mathbb{R}^D$ , we use  $\vec{\text{Poi}}(\vec{\lambda})$  to denote the joint distribution of  $D$  independent Poisson random variables:

$$\vec{\text{Poi}}(\vec{\lambda}) := (\text{Poi}(\vec{\lambda}_1), \text{Poi}(\vec{\lambda}_2), \dots, \text{Poi}(\vec{\lambda}_n)).$$

For a distribution  $\vec{U}$  on  $\mathbb{R}^D$ , we define the corresponding mixture of multi-dimensional Poisson distributions as follows:

$$\mathbb{E}[\vec{\text{Poi}}(\vec{U})] := \mathbb{E}_{\vec{\lambda} \leftarrow \vec{U}} \vec{\text{Poi}}(\vec{\lambda}).$$

For two random variables  $X$  and  $Y$  supported on  $\mathbb{R}^{\mathcal{M}}$ , we use  $X + Y$  to denote the random variable distributed as a sum of two independent samples from  $X$  and  $Y$ .

**Shuffling the Outputs of the Local Protocol.** Our first observation is that the analyzer for any local protocol computing **CountDistinct** should achieve the same accuracy if it only sees the histogram of the randomizers' outputs. This holds because only seeing the histogram of the outputs is equivalent to shuffling the outputs by a uniformly random permutation, which is in turn equivalent to shuffling the users in the dataset uniformly at random. Since shuffling the users in a dataset does not affect the number of distinct elements, it follows that only seeing the histogram does not affect the accuracy. Therefore, we only have to consider the histogram of the outputs of the local protocol computing **CountDistinct**. For a dataset  $W$ , we use  $\text{Hist}_R(W)$  to denote the distribution of the histogram with randomizer  $R$ .



**Poissonization Trick.** Given a distribution  $\mathcal{D}$  on  $\mathcal{M}$ , suppose we draw a sample  $m \leftarrow \text{Poi}(\lambda)$ , and then draw  $m$  samples from  $\mathcal{D}$ . If we use  $N$  to denote the random variable corresponding to the histogram of these  $m$  samples, it follows that each coordinate of  $N$  is independent, and  $N$  is distributed as  $\vec{\text{Poi}}(\lambda \vec{\mu})$ , where  $\vec{\mu}_i = \mathcal{D}_i$  for each  $i \in \mathcal{M}$ .

We can now apply the above trick to the context of local protocols (recall that by our first observation, we can focus on the histogram of the outputs). Suppose we build a dataset by drawing a sample  $m \leftarrow \text{Poi}(\lambda)$  and then adding  $m$  users with input  $z$ . By the above discussion, the corresponding histogram of the outputs with randomizer  $R$  is distributed as  $\vec{\text{Poi}}(\lambda \cdot R(z))$ , where  $R(z)$  is treated as an  $|\mathcal{M}|$ -dimensional vector corresponding to its probability distribution.

**Moment-Matching Random Variables.** Our next ingredient is the following construction of two moment-matching random variables used in [45]. Let  $L \in \mathbb{N}$  and  $\Lambda = \Theta(L^2)$ . There are two random variables  $U$  and  $V$  supported on  $\{0\} \cup [1, \Lambda]$ , such that  $\mathbb{E}[U] = \mathbb{E}[V] = 1$  and  $\mathbb{E}[U^j] = \mathbb{E}[V^j]$  for every  $j \in [L]$ . Moreover  $U_0 - V_0 > 0.9$ . That is,  $U$  and  $V$  have the same moments up to degree  $L$ , while the probabilities of them being zero differs significantly. We will set  $L = \log n$  and hence  $\Lambda = \Theta(\log^2 n)$ .

**Construction of Hard Distribution via Signal/Noise Decomposition.** Recalling that  $D = n/\log^5 n$ , we will construct two input distributions for  $\text{CountDistinct}_{n,D}$ .<sup>12</sup> A sample from both distributions consists of two parts: a signal part with  $D$  many users in expectation, and a noise part with  $n - D$  many users in expectation.

Formally, for a distribution  $W$  over  $\mathbb{R}^{\geq 0}$  and a subset  $E \subseteq [D]$ , the dataset distributions  $\mathcal{D}_{\text{signal}}^W$  and  $\mathcal{D}_{\text{noise}}^E$  are constructed as follows:

- $\mathcal{D}_{\text{signal}}^W$ : for each  $i \in [D]$ , we independently draw  $\lambda_i \leftarrow W$ , and  $n_i \leftarrow \text{Poi}(\lambda_i)$ , and add  $n_i$  many users with input  $i$ .
- $\mathcal{D}_{\text{noise}}^E$ : for each  $i \in E$ , we independently draw  $n_i \leftarrow \text{Poi}((n - D)/|E|)$ , and add  $n_i$  many users with input  $i$ .

We are going to fix a “good” subset  $E$  of  $[D]$  such that  $|E| \leq 0.02 \cdot D$  (we will later specify the other conditions for being “good”). Therefore, when it is clear from the context, we will use  $\mathcal{D}_{\text{noise}}$  instead of  $\mathcal{D}_{\text{noise}}^E$ .

Our two hard distributions are then constructed as  $\mathcal{D}^U := \mathcal{D}_{\text{signal}}^U + \mathcal{D}_{\text{noise}}$  and  $\mathcal{D}^V := \mathcal{D}_{\text{signal}}^V + \mathcal{D}_{\text{noise}}$ . Using the fact that  $\mathbb{E}[U] = \mathbb{E}[V] = 1$ , one can verify that there are  $D$  users in each of  $\mathcal{D}_{\text{signal}}^U$  and  $\mathcal{D}_{\text{signal}}^V$  in expectation. Similarly, one can also verify there are  $n - D$  users in  $\mathcal{D}_{\text{noise}}$  in expectation. Hence, both  $\mathcal{D}^U$  and  $\mathcal{D}^V$  have  $n$  users in expectation. In fact, the number of users from both distributions concentrates around  $n$ .

We now justify our naming of the signal/noise distributions. First, note that the number of distinct elements in the signal parts  $\mathcal{D}_{\text{signal}}^U$  and  $\mathcal{D}_{\text{signal}}^V$  concentrates around  $(1 - \mathbb{E}[e^{-U}]) \cdot D$  and  $(1 - \mathbb{E}[e^{-V}]) \cdot D$  respectively. By our condition that  $U_0 - V_0 > 0.9$ , it follows that the signal parts of  $\mathcal{D}^U$  and  $\mathcal{D}^V$  separates their numbers of distinct elements by at least  $0.4D$ . Second, note that although  $\mathcal{D}_{\text{noise}}$  has  $n - D \gg D$  many users in expectation, they are from the subset  $E$  of size less than  $0.02 \cdot n$ . Therefore, these users collectively cannot change the number of distinct elements by more than  $0.02 \cdot n$ , and the numbers of distinct elements in  $\mathcal{D}^U$  and  $\mathcal{D}^V$  are still separated by  $\Omega(D)$ .

<sup>12</sup> In fact, in our presentation the number of inputs in each dataset from our hard distributions will not be exactly  $n$ , but only concentrated around  $n$ . This issue can be easily resolved by throwing “extra” users in the dataset; we refer the reader to the full version for the details.

**Decomposition of Noise Part.** To establish the desired lower bound, it now suffices to show for the local randomizer  $R$ , it holds that  $\text{Hist}_R(\mathcal{D}^U)$  and  $\text{Hist}_R(\mathcal{D}^V)$  are very close in statistical distance. For  $W \in \{U, V\}$ , we can decompose  $\text{Hist}_R(\mathcal{D}^W)$  as

$$\text{Hist}_R(\mathcal{D}^W) = \sum_{i \in [D]} \vec{\text{Poi}}(W \cdot R(i)) + \sum_{i \in [E]} \vec{\text{Poi}}((n-D)/|E| \cdot R(i)).$$

By the additive property of Poisson distributions, letting  $\vec{v} = (n-D)/|E| \cdot \sum_{i \in [E]} R(i)$ , we have that  $\sum_{i \in [E]} \vec{\text{Poi}}((n-D)/|E| \cdot R(i)) = \vec{\text{Poi}}(\vec{v})$ .

Our key idea is to decompose  $\vec{v}$  carefully into  $D+1$  nonnegative vectors  $\vec{v}^{(0)}, \vec{v}^{(1)}, \dots, \vec{v}^{(D)}$ , such that  $\vec{v} = \sum_{i=0}^D \vec{v}^{(i)}$ . Then, for  $W \in \{U, V\}$ , we have

$$\text{Hist}_R(\mathcal{D}^W) = \vec{\text{Poi}}(\vec{v}^{(0)}) + \sum_{i \in [D]} \vec{\text{Poi}}(W \cdot R(i) + \vec{v}^{(i)}).$$

To show that  $\text{Hist}_R(\mathcal{D}^U)$  and  $\text{Hist}_R(\mathcal{D}^V)$  are close, it suffices to show that for each  $i \in [D]$ , it is the case that  $\vec{\text{Poi}}(U \cdot R(i) + \vec{v}^{(i)})$  and  $\vec{\text{Poi}}(V \cdot R(i) + \vec{v}^{(i)})$  are close. We show that they are close when  $\vec{v}^{(i)}$  is sufficiently large on every coordinate compared to  $R(i)$ .

► **Lemma 19.** *For each  $i \in [D]$ , and every  $\vec{\lambda} \in (\mathbb{R}^{\geq 0})^{\mathcal{M}}$ , if  $\vec{\lambda}_z \geq 2\Lambda^2 \cdot R(i)_z$  for every  $z \in \mathcal{M}$ , then<sup>13</sup>*

$$\|\mathbb{E}[\vec{\text{Poi}}(U \cdot R(i) + \vec{\lambda})] - \mathbb{E}[\vec{\text{Poi}}(V \cdot R(i) + \vec{\lambda})]\|_{TV} \leq \frac{1}{n^2}.$$

To apply Lemma 19, we simply set  $\vec{v}^{(i)} = (2\Lambda^2) \cdot R(i)$  and  $\vec{v}^{(0)} = \vec{v} - \sum_{i \in [D]} \vec{v}^{(i)}$ . Letting  $\vec{\mu} = \sum_{i \in [D]} R(i)$ , the requirement that  $\vec{v}^{(0)}$  has to be nonnegative translates to  $\vec{v}_z \geq 2\Lambda^2 \cdot \vec{\mu}_z$ , for each  $z \in \mathcal{M}$ .

**Construction of a Good Subset  $E$ .** So we want to pick a subset  $E \subseteq [D]$  of size at most  $0.02 \cdot D$  such that the corresponding  $\vec{v}^E = (n-D)/|E| \cdot \sum_{i \in [E]} R(i)$  satisfies  $\vec{v}_z^E \geq 2\Lambda^2 \cdot \vec{\mu}_z$  for each  $z \in \mathcal{M}$ . We will show that a simple random construction works with high probability: i.e., one can simply add each element of  $[D]$  to  $E$  independently with probability 0.01.

More specifically, for each  $z \in \mathcal{M}$ , we will show that with high probability  $\vec{v}_z^E \geq 2\Lambda^2 \cdot \vec{\mu}_z$ . Then the correctness of our construction follows from a union bound (and this step crucially uses the fact that  $|\mathcal{M}| \leq \text{poly}(n)$ ).

Now, let us fix a  $z \in \mathcal{M}$ . Let  $m^* = \max_{i \in [D]} R(i)_z$ . Since  $R$  is  $\ln(n/\log^7 n)$ -DP, it follows that  $\vec{v}_z \geq \frac{n-D}{n/\log^7 n} \cdot m^* \geq \frac{\log^7 n}{2} \cdot m^*$ . We consider the following two cases:

1. If  $m^* \geq \vec{\mu}_z / \log^2 n$ , we immediately get that  $\vec{v}_z \geq \log^5 n / 2 \cdot \vec{\mu}_z \geq 2\Lambda^2 \cdot \vec{\mu}_z$  (which uses the fact that  $\Lambda = \Theta(\log^2 n)$ ).
2. If  $m^* < \vec{\mu}_z / \log^2 n$ , then in this case, the mass  $\vec{\mu}_z$  is distributed over at least  $\log^2 n$  many components  $R(i)_z$ . Applying Hoeffding's inequality shows that with high probability over  $E$ , it is the case that  $\vec{v}_z^E \geq \Theta(n/D) \cdot \vec{\mu}_z \geq \Lambda^2 \cdot \vec{\mu}_z$  (which uses the fact that  $D = n/\log^5 n$ ).

See the full version for a formal argument and how to get rid of the assumption that  $|\mathcal{M}| \leq \text{poly}(n)$ .

<sup>13</sup> We use  $\|\mathcal{D}_1 - \mathcal{D}_2\|_{TV}$  to denote the total variation (aka statistical) distance between two distributions  $\mathcal{D}_1, \mathcal{D}_2$ .

**The Lower Bound.** From the above discussions, we get that

$$\|\text{Hist}_R(\mathcal{D}^U) - \text{Hist}_R(\mathcal{D}^V)\|_{TV} \leq \sum_{i=1}^D \|\mathbb{E}[\vec{\text{Poi}}(U \cdot R(i) + \vec{v}^{(i)})] - \mathbb{E}[\vec{\text{Poi}}(V \cdot R(i) + \vec{v}^{(i)})]\|_{TV} \leq 1/n.$$

Hence, the analyzer of the local protocol with randomizer  $R$  cannot distinguish  $\mathcal{D}^U$  and  $\mathcal{D}^V$ , and thus it cannot solve  $\text{CountDistinct}_{n,D}$  with error  $o(D)$  and 0.99 probability. See the full version for a formal argument and how to deal with the fact that there may not be exactly  $n$  users in dataset from  $\mathcal{D}^U$  or  $\mathcal{D}^V$ .

**Single-Message  $\text{DP}_{\text{shuffle}}^1$  Lower Bound.** To apply the above lower bound to  $\text{DP}_{\text{shuffle}}^1$  protocols, the natural idea is to resort to the connection between the  $\text{DP}_{\text{shuffle}}^1$  and  $\text{DP}_{\text{local}}$  models. In particular, [12] showed that  $(\varepsilon, \delta)$ - $\text{DP}_{\text{shuffle}}^1$  protocols are also  $(\varepsilon + \ln n, \delta)$ - $\text{DP}_{\text{local}}$ .

It may seem that the  $\ln n$  privacy guarantee is very close to the  $\ln n - O(\ln \ln n)$  one in Theorem 2. But surprisingly, it turns out (as was stated in Theorem 3) that there is a  $(\ln n + O(1))$ - $\text{DP}_{\text{local}}$  protocol solving  $\text{CountDistinct}_{n,n}$  (hence also  $\text{CountDistinct}_{n,D}$ ) with error  $O(\sqrt{n})$ . Hence, to establish the  $\text{DP}_{\text{shuffle}}^1$  lower bound (Theorem 4), we rely on the following stronger connection between  $\text{DP}_{\text{shuffle}}^1$  and  $\text{DP}_{\text{local}}$  protocols.

► **Lemma 20** (Simplification of Lemma 5). *For every  $\delta \leq 1/n^{\omega(1)}$ , if the randomizer  $R$  is  $(O(1), \delta)$ - $\text{DP}_{\text{shuffle}}^1$  on  $n$  users, then  $R$  is  $(\ln(n \log^2 n / \log \delta^{-1}), n^{-\omega(1)})$ - $\text{DP}_{\text{local}}$ .*

Setting  $\delta = 2^{-\log^k n}$  for a sufficiently large  $k$  and combining Lemma 20 and Theorem 2 gives us the desired lower bound against  $\text{DP}_{\text{shuffle}}^1$  protocols.

### 3.2 Lower Bounds for $\text{CountDistinct}$ and Selection via Dominated Protocols

We will first describe the proof ideas behind Theorem 1, which is restated below. Then, we apply the same proof technique to obtain lower bounds for **Selection** (the lower bound for **ParityLearning** is established similarly; see the full version for details).

► **Lemma 21** (Detailed Version of Theorem 1). *For  $\varepsilon = o(\ln n)$ , no  $(\varepsilon, o(1/n))$ -dominated protocol can solve  $\text{CountDistinct}$  with error  $o(n/e^\varepsilon)$ .*

**Hard Distributions for  $\text{CountDistinct}_{n,n}$ .** We now construct our hard instances for  $\text{CountDistinct}_{n,n}$ . For simplicity, we assume  $n = 2^D$  for an integer  $D$ , and identify the input space  $[n]$  with  $\{0, 1\}^D$  by a fixed bijection. Let  $\mathcal{U}_D$  be the uniform distribution over  $\{0, 1\}^D$ . For  $(\ell, s) \in [2] \times \{0, 1\}^D$ , we let  $\mathcal{D}_{\ell,s}$  be the uniform distribution on  $\{x \in \{0, 1\}^D : \langle x, s \rangle = \ell\}$ .

We also use  $\mathcal{D}_{\ell,s}^\alpha$  to denote the mixture of  $\mathcal{D}_{\ell,s}$  and  $\mathcal{U}_D$  which outputs a sample from  $\mathcal{D}_{\ell,s}$  with probability  $\alpha$  and a sample from  $\mathcal{U}_D$  with probability  $1 - \alpha$ .

For a parameter  $\alpha > 0$ , we consider the following two dataset distributions with  $n$  users:

- $\mathcal{W}^{\text{uniform}}$ : each user gets an i.i.d. input from  $\mathcal{U}_D$ . That is,  $\mathcal{W}^{\text{uniform}} := \mathcal{U}_D^{\otimes n}$ .
- $\mathcal{W}^\alpha$ : to sample a dataset from  $\mathcal{W}^\alpha$ , we first draw  $(\ell, s)$  from  $[2] \times \{0, 1\}^D$  uniformly at random, then each user gets an i.i.d. input from  $\mathcal{D}_{\ell,s}^\alpha$ . Formally,  $\mathcal{W}^\alpha := \mathbb{E}_{(\ell,s) \leftarrow [2] \times \{0,1\}^D} (\mathcal{D}_{\ell,s}^\alpha)^{\otimes n}$ .

Since for every  $\ell, s$ , it holds that  $|\text{supp}(\mathcal{D}_{\ell,s}^1)| \leq n/2$ , the number of distinct elements from any dataset in  $\mathcal{W}^1$  is at most  $n/2$ . On the other hand, since  $\mathcal{U}_D$  is a uniform distribution over  $n$  elements, a random dataset from  $\mathcal{W}^{\text{uniform}} = \mathcal{W}^0$  has roughly  $(1 - e^{-1}) \cdot n > n/2$  distinct

elements with high probability. Hence, the expected number of distinct elements of datasets from  $\mathcal{W}^\alpha$  is controlled by the parameter  $\alpha$ . A simple but tedious calculation shows that it is approximately  $(1 - e^{-1} \cdot \cosh(\alpha)) \cdot n$ , which can be approximated by  $(1 - e^{-1} \cdot (1 + \alpha^2)) \cdot n$  for  $n^{-0.1} < \alpha < 0.01$ . Hence, any protocol solving **CountDistinct** with error  $o(\alpha^2 n)$  should be able to distinguish between the above two distributions. Our goal is to show that this is impossible for  $(\varepsilon, o(1/n))$ -dominated protocols.

**Bounding KL Divergence for Dominated Protocols.** Our next step is to upper-bound the statistical distance  $\|\text{Hist}_R(\mathcal{W}^{\text{uniform}}) - \text{Hist}_R(\mathcal{W}^\alpha)\|_{TV}$ . As in previous work [41, 27, 23], we may upper-bound the KL divergence instead. By the convexity and chain-rule properties of KL divergence, it follows that

$$\begin{aligned} \text{KL}(\text{Hist}_R(\mathcal{W}^\alpha) \parallel \text{Hist}_R(\mathcal{W}^{\text{uniform}})) &\leq \mathbb{E}_{(\ell,s) \leftarrow [2] \times \{0,1\}^D} \text{KL}(R(\mathcal{D}_{\ell,s}^\alpha)^{\otimes n} \parallel R(\mathcal{U}_D)^{\otimes n}) \\ &= n \cdot \mathbb{E}_{(\ell,s) \leftarrow [2] \times \{0,1\}^D} \text{KL}(R(\mathcal{D}_{\ell,s}^\alpha) \parallel R(\mathcal{U}_D)). \end{aligned} \quad (1)$$

**Bounding the Average KL Divergence between a Family and a Single Distribution.** We are now ready to introduce our general tool for bounding average KL divergence quantities like (1). We first set up some notation. Let  $\mathcal{I}$  be an index set and  $\{\lambda_v\}_{v \in \mathcal{I}}$  be a family of distributions on  $\mathcal{X}$ , let  $\pi$  be a distribution on  $\mathcal{I}$ , and  $\mu$  be a distribution on  $\mathcal{X}$ . For simplicity, we assume that for every  $x \in \mathcal{X}$  and  $v \in \mathcal{I}$ , it holds that  $(\lambda_v)_x \leq 2 \cdot \mu_x$  (which is true for  $\{\mathcal{D}_{\ell,s}^\alpha\}_{(\ell,s) \in [2] \times \{0,1\}^D}$  and  $\mathcal{U}_D$ ).

► **Theorem 22.** *Let  $W: \mathbb{R} \rightarrow \mathbb{R}$  be a concave function such that for all functions  $\psi: \mathcal{X} \rightarrow \mathbb{R}^{\geq 0}$  satisfying  $\psi(\mu) \leq 1$ , it holds that*

$$\mathbb{E}_{v \leftarrow \pi} [(\psi(\lambda_v) - \psi(\mu))^2] \leq W(\|\psi\|_\infty).$$

*Then for an  $(\varepsilon, \delta)$ -dominated randomizer  $R$ , it follows that*

$$\mathbb{E}_{v \leftarrow \pi} [\text{KL}(R(\lambda_v) \parallel R(\mu))] \leq O(W(2e^\varepsilon) + \delta).$$

Similar theorems are proved in the previous work [17, 18, 41, 23] but only for locally private randomizers. Theorem 22 can be seen as a generalization of these previous results to dominated protocols.

**Bounding (1) via Fourier Analysis.** To apply Theorem 22, for  $f: \mathcal{X} \rightarrow \mathbb{R}^{\geq 0}$  with  $f(\mathcal{U}_D) = \mathbb{E}_{x \in \{0,1\}^D} [f(x)] \leq 1$ , we want to bound

$$\mathbb{E}_{(\ell,s) \leftarrow [2] \times \{0,1\}^D} [(f(\mathcal{D}_{\ell,s}^\alpha) - f(\mathcal{U}_D))^2] = \mathbb{E}_{s \in \{0,1\}^D} \alpha^2 \cdot \hat{f}(s)^2.$$

By Parseval's Identity (see Lemma 17),

$$\sum_{s \in \{0,1\}^D} \hat{f}(s)^2 = \mathbb{E}_{x \in \{0,1\}^D} f(x)^2 \leq f(\mathcal{U}_D) \cdot \|f\|_\infty \leq \|f\|_\infty.$$

Therefore, we can set  $W(L) := \alpha^2 \cdot \frac{L}{2^D}$ , and apply Theorem 22 to obtain

$$\mathbb{E}_{(\ell,s) \leftarrow [2] \times \{0,1\}^D} \text{KL}(R(\mathcal{D}_{\ell,s}^\alpha) \parallel R(\mathcal{U}_D)) \leq O(\alpha^2 \cdot e^\varepsilon / n + \delta).$$

We set  $\alpha$  such that  $\alpha^2 = c/e^\varepsilon$  for a sufficiently small constant  $c$  and note that  $\delta = o(1/n)$ . It follows that

$$\text{KL}(\text{Hist}_R(\mathcal{W}^\alpha) \parallel \text{Hist}_R(\mathcal{W}^{\text{uniform}})) \leq 0.01,$$

and therefore

$$\|\text{Hist}_R(\mathcal{W}^\alpha) - \text{Hist}_R(\mathcal{W}^{\text{uniform}})\|_{TV} \leq 0.1$$

by Pinsker's inequality. Hence, we conclude that  $(\varepsilon, o(1/n))$ -dominated protocols cannot solve  $\text{CountDistinct}_{n,n}$  with error  $o(n/e^\varepsilon)$ , completing the proof of Lemma 21. Now Theorem 1 follows from Lemma 21 and the fact that  $(\varepsilon, \delta)$ - $\text{DP}_{\text{local}}$  protocols are also  $(\varepsilon, \delta)$ -dominated.

**Lower Bounds for Selection against Multi-Message  $\text{DP}_{\text{shuffle}}$  Protocols.** Now we show how to apply Theorem 22 and Lemma 20 to prove lower bounds for Selection. For  $(\ell, j) \in [2] \times [D]$ , let  $\mathcal{D}_{\ell,j}$  be the uniform distribution on all length- $D$  binary strings with  $j$ th bit being  $\ell$ . Recall that  $\mathcal{U}_D$  is the uniform distribution on  $\{0, 1\}^D$ . Again we aim to upper-bound the average-case KL divergence  $\mathbb{E}_{(\ell,j) \leftarrow [2] \times [D]} \text{KL}(R(\mathcal{D}_{\ell,j}) \parallel R(\mathcal{U}_D))$ .

To apply Theorem 22, for  $f: \mathcal{X} \rightarrow \mathbb{R}^{\geq 0}$  with  $f(\mathcal{U}_D) = \mathbb{E}_{x \in \{0,1\}^D} [f(x)] \leq 1$ , we want to bound

$$\mathbb{E}_{(\ell,j) \leftarrow [2] \times [D]} [(f(\mathcal{D}_{\ell,j}) - f(\mathcal{U}_D))^2] = \mathbb{E}_{j \in [D]} \hat{f}(\{j\})^2.$$

By Lemma 16, it is the case that

$$\sum_{j \in [D]} \hat{f}(\{j\})^2 \leq O(\log \|f\|_\infty).$$

Therefore, we can set  $W(L) := c_1 \cdot \frac{\log L}{D}$  for an appropriate constant  $c_1$ , and apply Theorem 22 to obtain

$$\mathbb{E}_{(\ell,j) \leftarrow [2] \times [D]} \text{KL}(R(\mathcal{D}_{\ell,j}) \parallel R(\mathcal{U}_D)) \leq O\left(\frac{\varepsilon}{D} + \delta\right).$$

Combining this with Lemma 20 completes the proof (see the full version for the details).

---

## References

- 1 John M Abowd. The US Census Bureau adopts differential privacy. In *KDD*, pages 2867–2867, 2018.
- 2 Kareem Amin, Matthew Joseph, and Jieming Mao. Pan-private uniformity testing. In *COLT*, pages 183–218, 2020.
- 3 Apple Differential Privacy Team. Learning with privacy at scale. *Apple Machine Learning Journal*, 2017.
- 4 Victor Balcer and Albert Cheu. Separating local & shuffled differential privacy via histograms. In *ITC*, pages 1:1–1:14, 2020.
- 5 Victor Balcer, Albert Cheu, Matthew Joseph, and Jieming Mao. Connecting robust shuffle privacy and pan-privacy. In *SODA*, 2021. [arXiv:2004.09481](#).
- 6 Borja Balle, James Bell, Adrià Gascón, and Kobbi Nissim. The privacy blanket of the shuffle model. In *CRYPTO*, pages 638–667, 2019.
- 7 Amos Beimel, Iftach Haitner, Kobbi Nissim, and Uri Stemmer. On the round complexity of the shuffle model. In *TCC*, 2020.

- 8 Andrea Bittau, Úlfar Erlingsson, Petros Maniatis, Ilya Mironov, Ananth Raghunathan, David Lie, Mitch Rudominer, Ushasree Kode, Julien Tinnes, and Bernhard Seefeld. Prochlo: Strong privacy for analytics in the crowd. In *SOSP*, pages 441–459, 2017.
- 9 Joshua Brody, Amit Chakrabarti, Ranganath Kondapally, David P Woodruff, and Grigory Yaroslavtsev. Beyond set disjointness: the communication complexity of finding the intersection. In *PODC*, pages 106–113, 2014.
- 10 Mark Bun, Jelani Nelson, and Uri Stemmer. Heavy hitters and the structure of local privacy. *TALG*, 15(4):1–40, 2019.
- 11 TH Hubert Chan, Elaine Shi, and Dawn Song. Optimal lower bound for differentially private multi-party aggregation. In *ESA*, pages 277–288, 2012.
- 12 Albert Cheu, Adam D. Smith, Jonathan Ullman, David Zeber, and Maxim Zhilyaev. Distributed differential privacy via shuffling. In *EUROCRYPT*, pages 375–403, 2019. [arXiv:1808.01394](#).
- 13 Albert Cheu and Jonathan Ullman. The limits of pan privacy and shuffle privacy for learning and estimation. *arXiv*, 2020. [arXiv:2009.08000](#).
- 14 Seung Geol Choi, Dana Dachman-Soled, Mukul Kulkarni, and Arkady Yerukhimovich. Differentially-private multi-party sketching for large-scale statistics. *PoPETs*, 3:153–174, 2020.
- 15 Damien Desfontaines, Andreas Lochbihler, and David Basin. Cardinality estimators do not preserve privacy. *PoPETs*, 2019(2):26–46, 2019.
- 16 Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. Collecting telemetry data privately. In *NIPS*, pages 3571–3580, 2017.
- 17 John C Duchi, Michael I Jordan, and Martin J Wainwright. Local privacy and statistical minimax rates. In *FOCS*, pages 429–438, 2013.
- 18 John C. Duchi and Feng Ruan. The right complexity measure in locally private estimation: It is not the Fisher information. *arXiv*, 2018. [arXiv:1806.05756](#).
- 19 Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In *EUROCRYPT*, pages 486–503, 2006. [doi:10.1007/11761679\\_29](#).
- 20 Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam D. Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, pages 265–284, 2006. [doi:10.1007/11681878\\_14](#).
- 21 Cynthia Dwork, Moni Naor, Toniann Pitassi, Guy N Rothblum, and Sergey Yekhanin. Pan-private streaming algorithms. In *ICS*, pages 66–80, 2010.
- 22 Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3-4):211–407, 2014. [doi:10.1561/04000000042](#).
- 23 Alexander Edmonds, Aleksandar Nikolov, and Jonathan Ullman. The power of factorization mechanisms in local and central differential privacy. In *STOC*, pages 425–438, 2020. [doi:10.1145/3357713.3384297](#).
- 24 Úlfar Erlingsson, Vitaly Feldman, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Abhradeep Thakurta. Amplification by shuffling: From local to central differential privacy via anonymity. In *SODA*, pages 2468–2479, 2019.
- 25 Úlfar Erlingsson, Vasyli Pihur, and Aleksandra Korolova. RAPPOR: Randomized aggregatable privacy-preserving ordinal response. In *CCS*, pages 1054–1067, 2014.
- 26 Badih Ghazi, Noah Golowich, Ravi Kumar, Pasin Manurangsi, Rasmus Pagh, and Ameya Velingker. Pure differentially private summation from anonymous messages. In *ITC*, pages 15:1–15:23, 2020.
- 27 Badih Ghazi, Noah Golowich, Ravi Kumar, Rasmus Pagh, and Ameya Velingker. On the power of multiple anonymous messages. *IACR Cryptol. ePrint Arch.*, 2019:1382, 2019. URL: <https://eprint.iacr.org/2019/1382>.
- 28 Andy Greenberg. Apple’s “differential privacy” is about collecting your data – but not your data. *Wired*, June, 13, 2016.

- 29 Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Cryptography from anonymity. In *FOCS*, pages 239–248, 2006.
- 30 Daniel M Kane, Jelani Nelson, and David P Woodruff. An optimal algorithm for the distinct elements problem. In *PODS*, pages 41–52, 2010.
- 31 Shiva Prasad Kasiviswanathan, Homin K Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. What can we learn privately? *SICOMP*, 40(3):793–826, 2011.
- 32 Michael Kearns. Efficient noise-tolerant learning from statistical queries. *JACM*, 45(6):983–1006, 1998.
- 33 Andrew McGregor, Ilya Mironov, Toniann Pitassi, Omer Reingold, Kunal Talwar, and Salil Vadhan. The limits of two-party differential privacy. In *FOCS*, pages 81–90, 2010.
- 34 Andrew McGregor, Ilya Mironov, Toniann Pitassi, Omer Reingold, Kunal Talwar, and Salil P. Vadhan. The limits of two-party differential privacy. *ECCC*, 18:106, 2011. URL: <http://eccc.hpi-web.de/report/2011/106>.
- 35 Frank McSherry and Kunal Talwar. Mechanism design via differential privacy. In *FOCS*, pages 94–103, 2007.
- 36 Darakhshan Mir, Shan Muthukrishnan, Aleksandar Nikolov, and Rebecca N Wright. Pan-private algorithms via statistics on sketches. In *PODS*, pages 37–48, 2011.
- 37 Ryan O’Donnell. *Analysis of Boolean Functions*. Cambridge University Press, 2014. URL: <http://www.cambridge.org/de/academic/subjects/computer-science/algorithmics-complexity-computer-algebra-and-computational-g/analysis-boolean-functions>.
- 38 Rasmus Pagh and Nina Mesing Stausholm. Efficient differentially private  $f_0$  linear sketching. *arXiv*, 2020. [arXiv:2001.11932](https://arxiv.org/abs/2001.11932).
- 39 Stephen Shankland. How Google tricks itself to protect Chrome user privacy. *CNET*, October, 2014.
- 40 Thomas Steinke and Jonathan Ullman. Tight lower bounds for differentially private selection. In *FOCS*, pages 552–563, 2017.
- 41 Jonathan Ullman. Tight lower bounds for locally differentially private selection. *arXiv*, 2018. [arXiv:1802.02638](https://arxiv.org/abs/1802.02638).
- 42 Salil Vadhan. The complexity of differential privacy. In *Tutorials on the Foundations of Cryptography*, pages 347–450. Springer, 2017.
- 43 Gregory Valiant and Paul Valiant. Estimating the unseen: Improved estimators for entropy and other properties. *JACM*, 64(6):37:1–37:41, 2017. doi:10.1145/3125643.
- 44 Stanley L Warner. Randomized response: A survey technique for eliminating evasive answer bias. *JASA*, 60(309):63–69, 1965.
- 45 Yihong Wu and Pengkun Yang. Chebyshev polynomials, moment matching, and optimal estimation of the unseen. *The Annals of Statistics*, 47(2):857–883, 2019.



# A Generalized Matching Reconfiguration Problem

Noam Solomon

Immunai, New York, NY, USA

noam@immunai.com

Shay Solomon

School of Electrical Engineering, Tel Aviv University, Israel

shayso@tauex.tau.ac.il

---

## Abstract

The goal in *reconfiguration problems* is to compute a *gradual transformation* between two feasible solutions of a problem such that all intermediate solutions are also feasible. In the *Matching Reconfiguration Problem* (MRP), proposed in a pioneering work by Ito et al. from 2008, we are given a graph  $G$  and two matchings  $M$  and  $M'$ , and we are asked whether there is a sequence of matchings in  $G$  starting with  $M$  and ending at  $M'$ , each resulting from the previous one by either adding or deleting a single edge in  $G$ , without ever going through a matching of size  $< \min\{|M|, |M'|\} - 1$ . Ito et al. gave a polynomial time algorithm for the problem, which uses the Edmonds-Gallai decomposition.

In this paper we introduce a natural generalization of the MRP that depends on an integer parameter  $\Delta \geq 1$ : here we are allowed to make  $\Delta$  changes to the current solution rather than 1 at each step of the transformation procedure. There is always a valid sequence of matchings transforming  $M$  to  $M'$  if  $\Delta$  is sufficiently large, and naturally we would like to minimize  $\Delta$ . We first devise an optimal transformation procedure for unweighted matching with  $\Delta = 3$ , and then extend it to weighted matchings to achieve asymptotically optimal guarantees. The running time of these procedures is linear.

We further demonstrate the applicability of this generalized problem to dynamic graph matchings. In this area, the number of changes to the maintained matching per update step (the *recourse bound*) is an important quality measure. Nevertheless, the *worst-case* recourse bounds of almost all known dynamic matching algorithms are prohibitively large, much larger than the corresponding update times. We fill in this gap via a surprisingly simple black-box reduction: Any dynamic algorithm for maintaining a  $\beta$ -approximate maximum cardinality matching with update time  $T$ , for any  $\beta \geq 1$ ,  $T$  and  $\varepsilon > 0$ , can be *transformed* into an algorithm for maintaining a  $(\beta(1 + \varepsilon))$ -approximate maximum cardinality matching with update time  $T + O(1/\varepsilon)$  and worst-case recourse bound  $O(1/\varepsilon)$ . This result generalizes for approximate maximum weight matching, where the update time and worst-case recourse bound grow from  $T + O(1/\varepsilon)$  and  $O(1/\varepsilon)$  to  $T + O(\psi/\varepsilon)$  and  $O(\psi/\varepsilon)$ , respectively;  $\psi$  is the graph *aspect-ratio*. We complement this positive result by showing that, for  $\beta = 1 + \varepsilon$ , the worst-case recourse bound of any algorithm produced by our reduction is optimal. As a corollary, several key dynamic approximate matching algorithms – with poor worst-case recourse bounds – are strengthened to achieve near-optimal worst-case recourse bounds with no loss in update time.

**2012 ACM Subject Classification** Theory of computation → Graph algorithms analysis

**Keywords and phrases** Dynamic algorithms, graph matching, reconfiguration problem, recourse bound

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.57

**Related Version** A full version of the paper is available at <https://arxiv.org/pdf/1803.05825.pdf>.

**Funding** *Shay Solomon*: Partially supported by the Israel Science Foundation grant No.1991/19 and by Len Blavatnik and the Blavatnik Family foundation.

**Acknowledgements** The authors thank Thatchaphol Saranurak for fruitful discussions.



© Noam Solomon and Shay Solomon;

licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 57; pp. 57:1–57:20

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

## 1 Introduction

The study of graph algorithms is mostly concerned with the measure of *(static) runtime*. Given a graph optimization problem, the standard objective is to design a fast (possibly approximation) algorithm, and ideally complement it with a matching lower bound on the runtime of any (approximation) algorithm for solving the problem. As an example, computing (from scratch) a 2-approximate minimum vertex cover (VC) can be done trivially in linear time, whereas a better-than-2 approximation for the minimum VC cannot be computed in polynomial time under the unique games conjecture [46].

The current paper is motivated by a natural need arising in networks that are prone to temporary or permanent changes. Such changes are sometimes part of the normal behavior of the network, as in *dynamic networks*, but changes could also be the result of unpredictable failures of nodes and edges, particularly in *faulty networks*. Consider a large-scale network  $G = (V, E, w)$  for which we need to solve, perhaps approximately, some graph optimization problem, and the underlying solution (e.g., a maximum matching) is being used for some practical purpose (e.g., scheduling in packet switches) throughout a long time span. If the network changes over time, the quality of the used solution may degrade until it is too poor to be used in practice and it may even become infeasible.

Instead of the standard objectives of optimization, the questions that arise here concern *reoptimization*: Can we “efficiently” transform one given solution (the *source*) to another one (the *target*) under “real-life constraints”? The efficiency of the *transformation procedure* could be measured in terms of running time, but in some applications making even small changes to the currently used solution may incur huge costs, possibly much higher than the runtime cost of computing (from scratch) a better solution; we shall use “procedure” and “process” interchangeably. In particular, this is often the case whenever the edges of the currently used solution are “hard-wired” in some physical sense, as in road networks. Various real-life constraints or objectives may be studied; the one we focus on in this work is that at any step (or every few steps) throughout the transformation process the current solution should be both feasible and of quality no worse (by much) than that of either the source or target solutions. This constraint is natural as it might be prohibitively expensive or even impossible to carry out the transformation process *instantaneously*. Instead, the transformation can be broken into *phases* each performing  $\leq \Delta$  changes to the transformed solution, where  $\Delta \geq 1$  is some parameter, so that the solution obtained at the end of each phase – to be used instead of the source solution – is both feasible and of quality no (much) worse than either the source or target. The transformed solution is to eventually coincide with the target solution.

The arising *reoptimization* meta-problem generalizes the well-studied framework of *reconfiguration problems*, which we discuss in Section 1.1. It is interesting from both practical and theoretical perspectives, since even the most basic and well-understood optimization problems become open in this setting. E.g., for the VC problem, *given* a better-than-2 approximate target VC, can we transform to it from any source VC subject to the above constraints? This is an example for a problem that is computationally hard in the standard sense but might be easy from a reoptimization perspective. In contrast, perhaps computationally easy problems, such as approximate maximum matching, are hard from a reoptimization perspective?

This meta-problem captures tension between (1) the *global* objective of transforming one global solution to another, and (2) the *local* objective of transforming *gradually* while having a feasible and high quality solution throughout the process. A similar tension is captured by various models of computation that involve locality, including dynamic graph algorithms,

distributed computing, property testing and local computation algorithms (LCA). The study of the meta-problem presented here could borrow from these related research fields, but, more importantly, we anticipate that it will also contribute to them; indeed, we present here an application of this meta-problem to dynamic graph algorithms.

## 1.1 Graph Reconfiguration

The framework of *reconfiguration problems* has been subject to growing interest in recent years. The term *reconfiguration* was coined in the work of Ito et al. [41], which unified earlier related problems and terminology (see, e.g., [40, 31, 22]) into a single framework. The general goal is to compute a *transformation* between two feasible solutions of a problem such that all intermediate solutions are also feasible, where each pair of consecutive solutions need to be *adjacent* under a fixed polynomially testable symmetric adjacency relation on the set of feasible solutions. Such a transformation arises naturally in many contexts, such as solving puzzles, motion planning, questions of evolvability (can genotype evolve into another one via individual “adjacent” mutations?), and similarity of DNA sequences in computational genomics and particularly gene editing, which is among the hottest scientific topics these days; see the surveys of [58, 51] for further details. In most previous work, two solutions are called adjacent if their symmetric difference has size 1. The most well-studied problem under this framework is graph matching. For brevity, we shall only discuss here papers on graph matching; see the surveys [58, 51] for discussions on other problems.

In the *Matching Reconfiguration Problem* (MRP), proposed in [41], we are given a graph  $G$  and two matchings  $M$  and  $M'$ , and we are asked whether there is a sequence of matchings in  $G$  starting with  $M$  and ending at  $M'$ , each resulting from the previous one by either adding or deleting a *single edge* in  $G$ , without ever going through a matching of size  $< \min\{|M|, |M'|\} - 1$ . Ito et al. gave a polynomial time algorithm for the problem, which uses the Edmonds-Gallai decomposition. In particular, in some cases such a transformation does not exist, and much of the difficulty is in the decision problem (decide if exists or not). The problem of generalizing this algorithm for weighted matchings was proposed as an open problem in [41], and remained open to date, partially since the algorithm of [41] for unweighted matchings already relies on a rather intricate decomposition. The work of [41] triggered interesting followups on MRP [44, 42, 39, 43, 21, 27]. In all these followups, the symmetric difference between two adjacent matchings is rather strict: it is fixed by either 1 or 2 in [44, 42, 39, 27], whereas in the context of perfect matchings the symmetric difference is an alternating cycle of length 4 [21, 43]. Perhaps since the symmetric difference in all the previous work is so strict, the goal was polynomial-time algorithms and hardness for the problem. The natural generalization of parameterizing the symmetric difference by an arbitrary  $\Delta, \Delta \geq 1$  – as in our *reoptimization meta-problem*, was not studied in prior work.

## 1.2 Our contribution

We study two fundamental graph matching problems under the aforementioned meta-problem: (approximate) maximum cardinality matching (MCM) and maximum weight matching (MWM). Our meta-problem is, in fact, inherently different than the original MRP. We are not interested in the decision version of the problem – we take  $\Delta$  to be large enough so that a transformation is *guaranteed* to exist. Thus we shift the focus from per-instance optimization to *existential optimization*, and our goal is to optimize  $\Delta$  so that any source matching can be transformed to any target matching by performing at most  $\Delta$  changes per step, while never reaching a *much worse* matching than either the source or the target along the way.

By “worse” we mean either in terms of size or weight, and we must indeed do a bit worse in some cases even for large  $\Delta$ ; the original MRP formulation for unweighted matchings allows to go down by 1 unit of size, and this slack is required also for large  $\Delta$ . For weighted graphs, naturally, a bigger slack is required. For both unweighted and weighted matchings, we provide transformation procedures with near-optimal guarantees and linear running time. Our results are summarized next; the transformation for approximate MWM (Theorem 2) is the most technically challenging part of this work.

► **Theorem 1 (MCM).** *For any source and target matchings  $\mathcal{M}$  and  $\mathcal{M}'$ , one can transform  $\mathcal{M}$  into (a possibly superset of)  $\mathcal{M}'$  via a sequence of phases consisting of  $\leq 3$  operations each (i.e.,  $\Delta = 3$ ), such that the matching at the end of each phase throughout this transformation is a valid matching for  $G$  of size  $\geq \min\{|\mathcal{M}|, |\mathcal{M}'| - 1\}$ . The runtime of this transformation procedure is  $O(|\mathcal{M}| + |\mathcal{M}'|)$ .*

► **Theorem 2 (MWM).** *For any source and target matchings  $\mathcal{M}$  and  $\mathcal{M}'$  with  $w(\mathcal{M}') > w(\mathcal{M})$ , and any  $\varepsilon > 0$ , one can transform  $\mathcal{M}$  into (a possibly superset of)  $\mathcal{M}'$  via a sequence of phases consisting of  $O(\frac{1}{\varepsilon})$  operations each (i.e.,  $\Delta = O(\frac{1}{\varepsilon})$ ), such that the matching obtained at the end of each phase throughout this transformation is a valid matching for  $G$  of weight  $\geq \max\{w(\mathcal{M}) - W, (1 - \varepsilon)w(\mathcal{M})\}$ , where  $W = \max_{e \in \mathcal{M}} w(e)$ . The runtime of this transformation procedure is  $O(|\mathcal{M}| + |\mathcal{M}'|)$ .*

► **Remark.** Theorem 2 assumes that  $w(\mathcal{M}') > w(\mathcal{M})$ . This assumption is made without loss of generality, since, if  $w(\mathcal{M}') \leq w(\mathcal{M})$ , we can apply a reversed transformation, so that the matching will always be of weight  $\geq \max\{w(\mathcal{M}') - W', (1 - \varepsilon)w(\mathcal{M}')\}$ , where  $W' = \max_{e \in \mathcal{M}'} w(e)$ .

In Section 5, we show that the guarantees provided by Theorems 1 and 2 are tight and asymptotically tight, respectively. Although our results may lead to the impression that there exists an efficient gradual transformation process to any graph optimization problem, we briefly discuss in Section 7 two trivial hardness results for the minimum VC and maximum independent set problems.

### 1.2.1 Application: A worst-case recourse bound for dynamic matching algorithms

In the standard *fully dynamic* setting we start from an empty graph  $G_0$  on  $n$  fixed vertices, and at each time step  $i$  a single edge  $(u, v)$  is either inserted to the graph  $G_{i-1}$  or deleted from it, resulting in graph  $G_i$ . In the *vertex update* setting we have vertex updates instead of edge updates; this setting was mostly studied for bipartite graphs [24, 25, 14].

The problem of maintaining a large matching in fully dynamic graphs was subject to intensive interest recently [52, 10, 50, 38, 53, 56, 18, 14, 29, 3, 32, 13]. The basic goal is to devise an algorithm for maintaining a large matching while keeping a tab on the *update time*, i.e., the time required to update the matching at each step. One may try to optimize the *amortized* (average) update time of the algorithm or its *worst-case* (maximum) update time, but both measures are defined with respect to a *worst-case* sequence of graphs.

“Maintaining” a matching with update time  $u_T$  translates into maintaining a data structure with update time  $u_T$ , which answers queries regarding the matching with a low, ideally constant, *query time*  $q_T$ . For a queried vertex  $v$  the answer is the only matched edge incident on  $v$ , or NULL if  $v$  is free, while for a queried edge  $e$  the answer is whether edge  $e$  is matched or not. All queries made following the same update step  $i$  should be answered *consistently* with respect to the same matching, hereafter the *output matching (at step  $i$ )*,

but queries made in the next update step  $i + 1$  may be answered with respect to a completely different matching. Thus even if the worst-case update time is low, the output matching may change significantly from one update step to the next; some natural scenarios where the output matching changes significantly per update step are discussed in Section 2.

The number of changes (or replacements) to the output matching per update step is an important measure of quality, sometimes referred to as the *recourse bound*, and the problem of optimizing it has received growing attention recently [33, 30, 37, 25, 26, 14, 15, 2, 47]. In applications such as job scheduling, web hosting, streaming content delivery, data storage and hashing, a replacement of a matched edge by another one may be costly, possibly much more than the runtime of computing these replacements. Moreover, when the recourse bound is low, one can efficiently *output* all the changes to the matching following every update step, which could be important in practical scenarios. In particular, a low recourse bound is important when the matching algorithm is used as a black-box subroutine inside a larger data structure or algorithm [17, 1]; see Section 2.3 for more details. We remark that the recourse bound (generally defined as the number of changes to some underlying structure per update step) has been well studied in the areas of dynamic and online algorithms for a plethora of optimization problems besides graph matching, such as MIS, set cover, Steiner tree, flow and scheduling; see [34, 36, 37, 9, 48, 6, 28, 35, 54], and the references therein.

There is a strong separation between the state-of-the-art amortized versus worst-case bounds for dynamic matching algorithms, in terms of both the time and the recourse bounds. A similar separation exists for numerous other problems, such as dynamic minimum spanning forest. In various practical scenarios, particularly in systems designed to provide real-time responses, a strict tab on the *worst-case update time* or on the *worst-case recourse bound* is crucial, thus an algorithm with a low amortized guarantee but a high worst-case guarantee is useless.

Despite the importance of the recourse bound measure, all known algorithms but one in the area of dynamic matchings (described in detail in the full version [55]; see Appendix C therein) provide no nontrivial worst-case recourse bounds whatsoever! The sole exception is an algorithm for maintaining a maximal matching with a worst-case update time  $O(\sqrt{m})$  and a constant recourse bound [50]. In this paper we fill in this gap via a surprisingly simple yet powerful black-box reduction (throughout  $\beta$ -MCM is a shortcut for  $\beta$ -approximate MCM):

► **Theorem 3.** *Any dynamic algorithm maintaining a  $\beta$ -MCM with update time  $T$ ,<sup>1</sup> for any  $\beta \geq 1$ ,  $T$  and  $\varepsilon > 0$ , can be transformed into an algorithm maintaining a  $(\beta(1 + \varepsilon))$ -MCM with update time  $T + O(1/\varepsilon)$  and worst-case recourse bound  $O(1/\varepsilon)$ . If the original time bound  $T$  is amortized/worst-case, so is the resulting time bound of  $T + O(1/\varepsilon)$ , while the recourse bound  $O(1/\varepsilon)$  always holds in the worst-case. This applies to the fully dynamic setting under edge and/or vertex updates.*

The proof of Theorem 3 is carried out in two steps. First we prove Theorem 1 by showing a simple transformation process for any two matchings  $\mathcal{M}$  and  $\mathcal{M}'$  of the same *static* graph. The second step of the proof, which is the key insight behind it, is that the gradual transformation process can be used *essentially as is* in fully dynamic graphs, while incurring a negligible loss to the size and approximation guarantee of the transformed matching.

In Section 6 we complement the positive result provided by Theorem 3 by proving that the recourse bound  $O(1/\varepsilon)$  is optimal (up to a constant factor) in the regime  $\beta = 1 + \varepsilon$ . In fact, the lower bound  $\Omega(1/\varepsilon)$  on the recourse bound holds even in the amortized sense

<sup>1</sup> Besides answering queries, we naturally assume that at any update step the entire matching can be output within time (nearly) linear in its size. All known algorithms satisfy this assumption.

and even in the incremental (insertion only) and decremental (deletion only) settings. For larger values of  $\beta$ , taking  $\varepsilon$  to be a sufficiently small constant gives rise to an approximation guarantee arbitrarily close to  $\beta$  with a constant recourse bound.

**A corollary of Theorem 3.** As a corollary of Theorem 3, all previous algorithms [38, 16, 53, 29, 3, 13, 59] with low worst-case update time are strengthened to achieve a worst-case recourse bound of  $O(1/\varepsilon)$  with only an additive overhead of  $O(1/\varepsilon)$  to the update time. (Some of these results were already strengthened in this way by using a previous version of the current work, which was posted to arXiv in 2018.) Since the update time of all these algorithms is larger than  $O(1/\varepsilon)$ , we get a recourse bound of  $O(1/\varepsilon)$  with no loss whatsoever in the update time! Moreover, all known algorithms with low amortized update time can be strengthened in the same way; e.g., in SODA'19 [32] (cf. [24]) it was shown that one can maintain a  $(1 + \varepsilon)$ -MCM in the incremental edge update setting with a constant (depending exponentially on  $\varepsilon$ ) amortized update time. While this algorithm yields a constant amortized recourse bound, no nontrivial (i.e.,  $o(n)$ ) worst-case recourse bound was known for this problem. Theorem 3 strengthens the result of [32] to maintain a  $(1 + \varepsilon)$ -MCM with a constant amortized update time and the optimal worst-case recourse bound of  $O(1/\varepsilon)$ . Since the recourse bound is an important measure of quality, this provides a significant contribution to the area of dynamic matching algorithms.

**Weighted matchings.** The result of Theorem 3 can be generalized for approximate MWM in graphs with bounded aspect ratio  $\psi$ , by using the much more intricate transformation provided by Theorem 2 (compared to Theorem 1), as summarized in the next theorem. (The *aspect ratio*  $\psi = \psi(G)$  of a weighted graph  $G = (V, E, w)$  is defined as  $\psi = \frac{\max_{e \in E} w(e)}{\min_{e \in E} w(e)}$ .)

► **Theorem 4.** *Any dynamic algorithm for maintaining a  $\beta$ -approximate MWM (shortly,  $\beta$ -MWM) with update time  $T$  in a dynamic graph with aspect ratio always bounded by  $\psi$ , for any  $\beta \geq 1$ ,  $T, \varepsilon > 0$  and  $\psi$ , can be transformed into an algorithm for maintaining a  $(\beta(1 + \varepsilon))$ -MWM with update time  $T + O(\psi/\varepsilon)$  and worst-case recourse bound  $O(\psi/\varepsilon)$ . If the original time bound  $T$  is amortized/worst-case, so is the resulting time bound of  $T + O(\psi/\varepsilon)$ , while the recourse bound  $O(\psi/\varepsilon)$  always holds in the worst-case. This applies to the fully dynamic setting under edge and/or vertex updates.*

**Scenarios with high recourse bounds.** There are various scenarios where high recourse bounds may naturally arise. In such scenarios our reductions (Theorems 3 and 4) can come into play to achieve low worst-case recourse bounds. Furthermore, although a direct application of our reductions may only hurt the update time, we demonstrate the usefulness of these reductions in achieving low update time bounds in some natural settings (where we might not care at all about recourse bounds); this, we believe, provides another strong motivation for our reductions. The details are provided in Section 2.

### 1.3 Related work

We discussed in Section 1.1 prior work on graph reconfiguration problems. Other than this line of work, there are also inherently different lines of work on “reoptimization”, which indeed can be interpreted broadly – there is an extensive and diverse body of research devoted to various notions of reoptimization; see [57, 8, 23, 20, 7, 11, 12, 54, 19], and the references therein. The common goal in all previous work on reoptimization (besides the one discussed in Section 1.1 on reconfiguration) is to (efficiently) compute an exact or approximate solution



to a new problem instance by using the solution for the old instance, where typically the solution for the new instance should be close to the original one under certain distance measure. Our work is inherently different than all such previous work, since our starting point is that *some solution to the new problem instance is given*, and the goal is to compute a *gradual transformation process* (subject to some constraints) between the two given solutions. Also, our work is inherently different than previous work on reconfiguration, as explained in Section 1.2.

## 1.4 Organization

We start (Section 2) with discussing some scenarios where high recourse bounds may naturally arise. We continue (Section 3) by describing a basic scheme for dynamic approximate matchings that was introduced in [38]. In Section 4.1 we present a simple transformation process for MCM in static graphs, thus proving Theorem 1. This result is generalized for MWM via a more intricate transformation process that proves Theorem 2, which is deferred to the full version [55] (see Appendix D therein) due to space constraints. These transformations, which apply to static graphs, are adapted to the fully dynamic setting in Sections 4.2 and 4.3, thus proving Theorems 3 and 4, respectively. The optimality of these transformations is discussed in Section 5. Our lower bound of  $\Omega(1/\varepsilon)$  on the recourse bound of  $(1 + \varepsilon)$ -MCMs is provided in Section 6. We conclude with a discussion in Section 7.

## 2 Scenarios with high recourse bounds

In this section we discuss some scenarios where high recourse bounds may naturally arise. In all such scenarios our reductions (Theorems 3 and 4) can come into play to achieve low worst-case recourse bounds; for clarity we focus in this discussion, sometimes implicitly, on large (unweighted) matching, but the entire discussion carries over with very minor changes to the generalized setting of weighted matchings.

Section 2.3 demonstrates that, although we *may not care at all about recourse bounds*, maintaining a large (weight) matching with a low update time requires in some cases the use of a dynamic matching algorithm with a low recourse bound; this is another situation where our reductions can come into play, but more than that, we believe that it provides an additional strong motivation for our reductions.

### 2.1 Randomized algorithms

**Multiple matchings.** Given a randomized algorithm for maintaining a large matching in a dynamic graph, it may be advantageous to run multiple instances of the algorithm (say  $\text{polylog}(n)$ ), since this may increase the chances that at least one of those instances provides a large matching with high probability (w.h.p.) at any point in time. Notice, however, that it is not the same matching that is guaranteed to be large throughout the entire update sequence, hence the ultimate algorithm (or data structure), which outputs the largest among the  $\text{polylog}(n)$  matchings, may need to switch between a pool of possibly very different matchings when going from one update step to the next. Thus even if the recourse bound of the given randomized algorithm is low, and so each of the maintained matchings changes gradually over time, we do not get any nontrivial recourse bound for the ultimate algorithm.

**Large matchings.** Sometimes the approximation guarantee of the given randomized algorithm holds w.h.p. only when the matching is sufficiently large. This is the case with the algorithm of [29] that achieves  $\text{polylog}(n)$  worst-case update time, where the approximation



guarantee of  $2 + \varepsilon$  holds w.h.p. only when the size of the matching is  $\Omega(\log^5 n / \varepsilon^4)$ . To perform efficiently, [29] also maintains a matching that is guaranteed to be maximal (and thus provide a 2-MCM) when the maximum matching size is smaller than  $\delta = O(\log^5 n / \varepsilon^4)$ , via a deterministic procedure with a worst-case update time of  $O(\delta)$ . The ultimate algorithm of [29] switches between the matching given by the randomized algorithm and that by the deterministic procedure, taking the larger of the two. Thus even if the recourse bounds of both the randomized algorithm and the deterministic procedure are low, the worst-case recourse bound of the ultimate algorithm, which might be of the order of the “large matching” threshold, could be very high. (The large matching threshold is the threshold on the matching size above which a high probability bound on the approximation guarantee holds.) In [29] the large matching threshold is  $\delta = O(\log^5 n / \varepsilon^4)$ , so the recourse bound is reasonably low. (This is not the bottleneck for the recourse bound of [29], as discussed next.) In general, however, the large matching threshold may be significantly higher than  $\text{polylog}(n)$ .

**Long update sequences.** For the probabilistic guarantees of a randomized dynamic algorithm to hold w.h.p., the update sequence must be of bounded length. In particular, polylogarithmic guarantees on the update time usually require that the length of the update sequence will be polynomially bounded. This is the case with numerous dynamic graph algorithms also outside the scope of graph matchings (cf. [45, 1]), and the basic idea is to partition the update sequence into sub-sequences of polynomial length each and to run a fresh instance of the dynamic algorithm in each sub-sequence. In the context of matchings, the algorithm of [29] uses this approach. Notice, however, that an arbitrary sub-sequence (other than the first) does not start from an empty graph. Hence, for the ultimate algorithm of [29] to provide a low worst-case update time, it has to gradually construct the graph at the beginning of each sub-sequence from scratch and maintain for it a new gradually growing matching, while re-using the “old” matching used for the previous sub-sequence throughout this gradual process. Once the gradually constructed graph coincides with the true graph, the ultimate algorithm switches from the old matching to the new one. (See [29] for further details.) While this approach guarantees that the worst-case update time of the algorithm is in check, it does not provide any nontrivial worst-case recourse bound.

## 2.2 From amortized to worst-case

There are techniques for transforming algorithms with low amortized bounds into algorithms with similar worst-case bounds. For approximate matchings, such a technique was first presented in [38]. Alas, the transformed algorithms do not achieve any nontrivial worst-case recourse bound; see Section 3 for details.

## 2.3 When low update time requires low recourse bound

When a dynamic matching algorithm is used as a black-box subroutine inside a larger data structure or algorithm, a low recourse bound of the algorithm used as a subroutine is needed for achieving a low update time for the larger algorithm. We next consider a natural question motivating this need; one may refer to [17, 1] for additional motivation.

► **Question 1.** *Given  $k$  dynamic matchings of a dynamic graph  $G$ , whose union is guaranteed to contain a large matching for  $G$  at any time, for an arbitrary parameter  $k$ , can we combine those  $k$  matchings into a large dynamic matching for  $G$  efficiently?*

This question may arise when there are physical limitations, such as memory constraints, e.g., as captured by MapReduce-style computation, where the edges of the graph are partitioned into  $k$  parties. More specifically, consider a fully dynamic graph  $G$  of huge scale, for which we want to maintain a large matching with low update time. The edges of the graph are dynamically partitioned into  $k$  parties due to memory constraints, each capable of maintaining a large matching for the graph induced by its own edges with low update time, and the only guarantee on those  $k$  dynamically changing matchings is the following global one: The *union* of the  $k$  matchings at any point in time *contains* a large matching for the entire dynamic graph  $G$ . (E.g., if we maintain at each update step the invariant that the edges of  $G$  are partitioned across the  $k$  parties uniformly at random, such a global guarantee can be provided via the framework of *composable randomized coresets* [49, 5, 4].)

This question may also arise when the input data set is noisy. Coping with noisy input usually requires *randomization*, which may lead to high recourse bounds as discussed in Section 2.1. Let us revisit the scenario where we run multiple instances of a randomized dynamic algorithm with low update time; denote the number of such instances by  $k$ . If the input is noisy, we may not be able to guarantee that at least one of the  $k$  maintained matchings is large w.h.p. at any point in time, as suggested in Section 2.1. A weaker, more reasonable assumption is that the union of those  $k$  matchings contains a large matching.

The key observation is that it is insufficient to maintain each of the  $k$  matchings with low update time, even in the worst-case, as each such matching may change significantly following a single update step, thereby changing significantly the union of those matchings. “Feeding” this union to *any* dynamic matching algorithm would result with poor update time bounds, even in the amortized sense. Consequently, to resolve Question 1, each of the  $k$  maintained matchings must change *gradually* over time, or in other words, the underlying algorithm(s) needed for maintaining those matchings should guarantee a low recourse bound. A low amortized/worst-case recourse bound of the underlying algorithm(s) translates into a low amortized/worst-case update time of the ultimate algorithm, provided of course that the underlying algorithm(s) for maintaining those  $k$  matchings, as well as the dynamic matching algorithm to which their union is fed, all achieve a low amortized/worst-case update time.

### 3 The scheme of [38]

This section provides a short overview of a basic scheme for dynamic approximate matchings from [38]. Although such an overview is not required for proving Theorems 3 and 4, it is instructive to provide it, as it shows that the scheme of [38] is insufficient for providing any nontrivial worst-case recourse bound. Also, the scheme of [38] exploits a basic *stability* property of matchings, which we use for proving Theorems 3 and 4, thus an overview of this scheme may facilitate the understanding of our proof.

#### 3.1 The amortization scheme of [38]

The *stability* property of matchings used in [38] is that the maximum matching size changes by at most 1 following each update step. Thus if we have a  $\beta$ -MCM, for any  $\beta \geq 1$ , the approximation guarantee of the matching will remain close to  $\beta$  throughout a long update sequence. Formally, the following lemma is a simple adaptation of Lemma 3.1 from [38]; its proof is given in Appendix E of the full version [55]. (Lemma 3.1 of [38] is stated for approximation guarantee  $1 + \varepsilon$  and for edge updates, whereas Lemma 5 here holds for any approximation guarantee and also for vertex updates.)

► **Lemma 5.** *Let  $\varepsilon' \leq 1/2$ . Suppose  $\mathcal{M}_t$  is a  $\beta$ -MCM for  $G_t$ , for any  $\beta \geq 1$ . For  $i = t, t+1, \dots, t + \lfloor \varepsilon' \cdot |\mathcal{M}_t| \rfloor$ , let  $\mathcal{M}_t^{(i)}$  denote the matching  $\mathcal{M}_t$  after removing from it all edges that got deleted during updates  $t+1, \dots, i$ . Then  $\mathcal{M}_t^{(i)}$  is a  $(\beta(1+2\varepsilon'))$ -MCM for  $G_i$ .*

For concreteness, we shall focus on the regime of approximation guarantee  $1 + \varepsilon$ , and sketch the argument of [38] for maintaining a  $(1 + \varepsilon)$ -MCM in fully dynamic graphs. (As Lemma 5 applies to any approximation guarantee  $\beta \geq 1 + \varepsilon$ , it is readily verified that the same argument carries over to any approximation guarantee.)

One can compute a  $(1 + \varepsilon/4)$ -MCM  $\mathcal{M}_t$  at a certain update step  $t$ , and then re-use the same matching  $\mathcal{M}_t^{(i)}$  throughout all update steps  $i = t, t+1, \dots, t' = t + \lfloor \varepsilon/4 \cdot |\mathcal{M}_t| \rfloor$  (after removing from it all edges that got deleted from the graph between steps  $t$  and  $i$ ). By Lemma 5, assuming  $\varepsilon \leq 1/2$ ,  $\mathcal{M}_t^{(i)}$  provides a  $(1 + \varepsilon)$ -MCM for all graphs  $G_i$ . Next compute a fresh  $(1 + \varepsilon/4)$ -MCM  $\mathcal{M}_{t'}$  following update step  $t'$  and re-use it throughout all update steps  $t', t'+1, \dots, t' + \lfloor \varepsilon/4 \cdot |\mathcal{M}_{t'}| \rfloor$ , and repeat. In this way the static time complexity of computing a  $(1 + \varepsilon)$ -MCM  $\mathcal{M}$  is *amortized* over  $1 + \lfloor \varepsilon/4 \cdot |\mathcal{M}| \rfloor = \Omega(\varepsilon \cdot |\mathcal{M}|)$  update steps. Note that the static computation time of an approximate matching is  $O(|\mathcal{M}| \cdot \alpha/\varepsilon^2)$ , where  $\alpha$  is the arboricity bound; refer to Appendix F in the full version [55]. (This bound on the static computation time was established in [53]; it reduces to  $O(|\mathcal{M}| \cdot \sqrt{m}/\varepsilon^2)$  and  $O(|\mathcal{M}| \cdot \Delta/\varepsilon^2)$  for general graphs and graphs of degree bounded by  $\Delta$ , respectively, which are the bounds provided by [38].)

### 3.2 A Worst-Case Update time

In the amortization scheme of [38] described above, a  $(1 + \varepsilon/4)$ -MCM  $\mathcal{M}$  is computed *from scratch*, and then being re-used throughout  $\lfloor \varepsilon/4 \cdot |\mathcal{M}| \rfloor$  additional update steps. The worst-case update time is thus the static computation time of an approximate matching, namely,  $O(|\mathcal{M}| \cdot \alpha/\varepsilon^2)$ . To improve the worst-case guarantee, the tweak used in [38] is to simulate the static approximate matching computation within a “time window” of  $1 + \lfloor \varepsilon/4 \cdot |\mathcal{M}| \rfloor$  consecutive update steps, so that following each update step the algorithm simulates only  $O(|\mathcal{M}| \cdot \alpha/\varepsilon^2)/(1 + \lfloor \varepsilon/4 \cdot |\mathcal{M}| \rfloor) = O(\alpha \cdot \varepsilon^{-3})$  steps of the static computation. During this time window the gradually-computed matching, denoted by  $\mathcal{M}'$ , is useless, so the previously-computed matching  $\mathcal{M}$  is re-used as the output matching. This means that each matching is re-used throughout a time window of twice as many update steps, hence the approximation guarantee increases from  $1 + \varepsilon$  to  $1 + 2\varepsilon$ , but we can reduce it back to  $1 + \varepsilon$  by a straightforward scaling argument. Note that the gradually-computed matching does not include edges that got deleted from the graph during the time window.

### 3.3 Recourse bounds

Consider an arbitrary time window used in the amortization scheme of [38], and note that the same matching is being re-used throughout the entire window. Hence there are no changes to the matching in the “interior” of the window except for those triggered by adversarial deletions, which may trigger at most one change to the matching per update step. On the other hand, at the start of any time window (except for the first), the output matching is switched from the old matching  $\mathcal{M}$  to the new one  $\mathcal{M}'$ , which may require  $|\mathcal{M}| + |\mathcal{M}'|$  replacements to the output matching at that time. Note that the amortized number of replacements per update step is quite low, being upper bounded by  $(|\mathcal{M}| + |\mathcal{M}'|)/(1 + \lfloor \varepsilon/4 \cdot |\mathcal{M}| \rfloor)$ . In the regime of approximation guarantee  $\beta = O(1)$ , we have  $|\mathcal{M}| = O(|\mathcal{M}'|)$ , hence the amortized recourse bound is bounded by  $O(1/\varepsilon)$ . For a general approximation guarantee  $\beta$ , the naive amortized recourse bound is  $O(\beta/\varepsilon)$ .

On the negative side, the worst-case recourse bound may still be as high as  $|\mathcal{M}| + |\mathcal{M}'|$ , even after performing the above tweak. Indeed, that tweak only causes the time windows to be twice longer, and it does not change the fact that once the computation of  $\mathcal{M}'$  finishes, the output matching is switched from the old matching  $\mathcal{M}$  to the new one  $\mathcal{M}'$  *instantaneously*, which may require  $|\mathcal{M}| + |\mathcal{M}'|$  replacements to the output matching at that time.

## 4 Proofs of Theorems 3 and 4

This section is mostly devoted (see Sections 4.1 and 4.2) to the proof of Theorem 3. At the end of this section (Section 4.3) we sketch the adjustments needed for deriving the result of Theorem 4, whose proof follows along similar lines to those of Theorem 3.

### 4.1 A simple transformation in static graphs

This section is devoted to the proof of Theorem 1, which provides the first step in the proof of Theorem 3. We remark that this proof can be viewed as a “warm up” to that of Theorem 2 for MWM, which is deferred to the full version [55] (see Appendix D therein), and is considerably more technically involved.

Let  $\mathcal{M}$  and  $\mathcal{M}'$  be two matchings for the same graph  $G$ . Our goal is to gradually transform  $\mathcal{M}$  into (a possibly superset of)  $\mathcal{M}'$  via a sequence of constant-time operations to be described next, each making at most 3 changes to the matching, such that the matching obtained at any point throughout this transformation process is a valid matching for  $G$  of size at least  $\min\{|\mathcal{M}|, |\mathcal{M}'| - 1\}$ . It is technically convenient to denote by  $\mathcal{M}^*$  the *transformed* matching, which is initialized as  $\mathcal{M}$  at the outset, and being gradually transformed into  $\mathcal{M}'$ ; we refer to  $\mathcal{M}$  and  $\mathcal{M}'$  as the *source* and *target* matchings, respectively. Each operation starts by adding a single edge of  $\mathcal{M}' \setminus \mathcal{M}^*$  to  $\mathcal{M}^*$  and then removing from  $\mathcal{M}^*$  the at most two edges incident on the newly added edge; thus at most 3 changes to the matching are made per operation. It is instructive to assume that  $|\mathcal{M}'| > |\mathcal{M}|$ , as the motivation for applying this transformation, which will become clear in Section 4.2, is to increase the matching size; in this case the size  $|\mathcal{M}^*|$  of the transformed matching  $\mathcal{M}^*$  never goes below the size  $|\mathcal{M}|$  of the source matching  $\mathcal{M}$ .

We say that an edge of  $\mathcal{M}' \setminus \mathcal{M}^*$  that is incident on at most one edge of  $\mathcal{M}^*$  is *good*, otherwise it is *bad*, being incident on two edges of  $\mathcal{M}^*$ . Since  $\mathcal{M}^*$  has to be a valid matching throughout the transformation process, adding a bad edge to  $\mathcal{M}^*$  must trigger the removal of two edges from  $\mathcal{M}^*$ . Thus if we keep adding bad edges to  $\mathcal{M}^*$ , the size of  $\mathcal{M}^*$  may halve throughout the transformation process. The following lemma shows that if all edges of  $\mathcal{M}' \setminus \mathcal{M}^*$  are bad, the transformed matching  $\mathcal{M}^*$  is at least as large as the target matching  $\mathcal{M}'$ .

► **Lemma 6.** *If all edges of  $\mathcal{M}' \setminus \mathcal{M}^*$  are bad, then  $|\mathcal{M}^*| \geq |\mathcal{M}'|$ .*

**Proof.** Consider a bipartite graph  $L \cup R$ , where each vertex in  $L$  corresponds to an edge of  $\mathcal{M}' \setminus \mathcal{M}^*$  and each vertex in  $R$  corresponds to an edge of  $\mathcal{M}^* \setminus \mathcal{M}'$ , and there is an edge between a vertex in  $L$  and a vertex in  $R$  iff the corresponding matched edges share a common vertex in the original graph. If all edges of  $\mathcal{M}' \setminus \mathcal{M}^*$  are bad, then any edge of  $\mathcal{M}' \setminus \mathcal{M}^*$  is incident on two edges of  $\mathcal{M}^*$ , and since  $\mathcal{M}'$  is a valid matching, those two edges cannot be in  $\mathcal{M}'$ . In other words, the degree of each vertex in  $L$  is exactly 2. Also, the degree of each vertex in  $R$  is at most 2, as  $\mathcal{M}'$  is a valid matching. It follows that  $|R| \geq |L|$ , or in other words  $|\mathcal{M}^* \setminus \mathcal{M}'| \geq |\mathcal{M}' \setminus \mathcal{M}^*|$ , yielding  $|\mathcal{M}^*| \geq |\mathcal{M}'|$ . ◀

The transformation process is carried out as follows. At the outset we initialize  $\mathcal{M}^* = \mathcal{M}$  and compute the sets  $\mathcal{G}$  and  $\mathcal{B}$  of good and bad edges in  $\mathcal{M}' \setminus \mathcal{M}^* = \mathcal{M}' \setminus \mathcal{M}$  within time  $O(|\mathcal{M}| + |\mathcal{M}'|)$  in the obvious way, and store them in doubly-linked lists. We keep mutual pointers between each edge of  $\mathcal{M}^*$  and its at most two incident edges in the corresponding linked lists  $\mathcal{G}$  and  $\mathcal{B}$ . Then we perform a sequence of operations, where each operation starts by adding an edge of  $\mathcal{M}' \setminus \mathcal{M}^*$  to  $\mathcal{M}^*$ , giving precedence to good edges (i.e., adding a bad edge to  $\mathcal{M}^*$  only when there are no good edges to add), and then removing from  $\mathcal{M}^*$  the at most two edges incident on the newly added edge. Following each such operation, we update the lists  $\mathcal{G}$  and  $\mathcal{B}$  of good and bad edges in  $\mathcal{M}' \setminus \mathcal{M}^*$  within constant time in the obvious way. This process is repeated until  $\mathcal{M}' \setminus \mathcal{M}^* = \emptyset$ , at which stage we have  $\mathcal{M}^* \supseteq \mathcal{M}'$ . Note that the number of operations performed before emptying  $\mathcal{M}' \setminus \mathcal{M}^*$  is bounded by  $|\mathcal{M}'|$ , since each operation removes at least one edge from  $\mathcal{M}' \setminus \mathcal{M}^*$ . It follows that the total runtime of the transformation process is bounded by  $O(|\mathcal{M}| + |\mathcal{M}'|)$ .

It is immediate that  $\mathcal{M}^*$  remains a valid matching throughout the transformation process, as we pro-actively remove from it edges that share a common vertex with new edges added to it. To complete the proof of Theorem 1 it remains to prove the following lemma.

► **Lemma 7.** *At any moment in time we have  $|\mathcal{M}^*| \geq \min\{|\mathcal{M}|, |\mathcal{M}'| - 1\}$ .*

**Proof.** Suppose for contradiction that the lemma does not hold, and consider the first time step  $t^*$  throughout the transformation process in which  $|\mathcal{M}^*| < \min\{|\mathcal{M}|, |\mathcal{M}'| - 1\}$ . Since initially  $|\mathcal{M}^*| = |\mathcal{M}|$  and as every addition of a good edge to  $\mathcal{M}^*$  triggers at most one edge removal from it, time step  $t^*$  must occur after an addition of a bad edge. Recall that a bad edge is added to  $\mathcal{M}^*$  only when there are no good edges to add. Just before this addition we have  $|\mathcal{M}^*| \geq |\mathcal{M}'|$  by Lemma 6, thus we have  $|\mathcal{M}^*| \geq |\mathcal{M}'| - 1$  after adding that edge to  $\mathcal{M}^*$  and removing the two edges incident on it from there, yielding a contradiction. ◀

► **Remark 8.** When  $|\mathcal{M}| < |\mathcal{M}'|$ , it is possible to gradually transform  $\mathcal{M}$  to  $\mathcal{M}'$  without ever being in deficit compared to the initial value of  $\mathcal{M}$ , i.e.,  $|\mathcal{M}^*| \geq |\mathcal{M}|$  throughout the transformation process. However, if  $|\mathcal{M}'| \leq |\mathcal{M}|$ , this no longer holds true; refer to Section 5.1 for more details.

## 4.2 The Fully Dynamic Setting

In this section we provide the second step in the proof of Theorem 3, showing that the simple transformation process described in Section 4.1 for static graphs can be generalized for the fully dynamic setting, thus completing the proof of Theorem 3.

Consider an arbitrary dynamic algorithm, Algorithm  $\mathcal{A}$ , for maintaining a  $\beta$ -MCM with an update time of  $T$ , for any  $\beta \geq 1$  and  $T$ . The matching maintained by Algorithm  $\mathcal{A}$ , denoted by  $\mathcal{M}_i^{\mathcal{A}}$ , for  $i = 1, 2, \dots$ , may change significantly following a single update step. All that is guaranteed by Algorithm  $\mathcal{A}$  is that it can update the matching following every update step within a time bound of  $T$ , either in the worst-case sense or in the amortized sense, following which queries regarding the matching can be answered in (nearly) constant time. Recall also that we assume that, for any update step  $i$ , the matching  $\mathcal{M}_i^{\mathcal{A}}$  provided by Algorithm  $\mathcal{A}$  at step  $i$  can be output within time (nearly) linear in the matching size.

Our goal is to output a matching  $\hat{\mathcal{M}} = \hat{\mathcal{M}}_i$ , for  $i = 1, 2, \dots$ , possibly very different from  $\mathcal{M}^{\mathcal{A}} = \mathcal{M}_i^{\mathcal{A}}$ , which changes very slightly from one update step to the next. To this end, the basic idea is to use the matching  $\mathcal{M}^{\mathcal{A}}$  provided by Algorithm  $\mathcal{A}$  at a certain update step, and then re-use it (gradually removing from it edges that get deleted from the graph) throughout a sufficiently long window of  $\Theta(\varepsilon \cdot |\mathcal{M}^{\mathcal{A}}|)$  consecutive update steps, while gradually transforming it into a larger matching, provided again by Algorithm  $\mathcal{A}$  at some later step.

The *gradual transformation process* is obtained by adapting the process described in Section 4.1 for static graphs to the fully dynamic setting. Next, we describe this adaptation. We assume that  $\beta = O(1)$ ; the case of a general  $\beta$  is addressed in Section 4.2.1.

Consider the beginning of a new time window, at some update step  $t$ . Denote the matching provided by Algorithm  $\mathcal{A}$  at that stage by  $\mathcal{M}' = \mathcal{M}_t^A$  and the matching output by our algorithm by  $\mathcal{M} = \tilde{\mathcal{M}}_t$ . Recall that the entire matching  $\mathcal{M}' = \mathcal{M}_t^A$  can be output in time (nearly) linear in its size, and we henceforth assume that  $\mathcal{M}'$  is given as a list of edges. (For concreteness, we assume that the time needed for storing the edges of  $\mathcal{M}'$  in an appropriate list is  $O(|\mathcal{M}'|)$ .) While  $\mathcal{M}'$  is guaranteed to provide a  $\beta$ -MCM at any update step, including  $t$ , the approximation guarantee of  $\mathcal{M}$  may be worse. Nevertheless, we will show (Lemma 9) that  $\mathcal{M}$  provides a  $(\beta(1 + 2\varepsilon'))$ -MCM for  $G_t$ . Under the assumption that  $\beta = O(1)$ , we thus have  $|\mathcal{M}| = O(|\mathcal{M}'|)$ . The length of the time window is  $W = \Theta(\varepsilon \cdot |\mathcal{M}|)$ , i.e., it starts at update step  $t$  and ends at update step  $t' = t + W - 1$ . During this time window, we gradually transform  $\mathcal{M}$  into (a possibly superset of)  $\mathcal{M}'$ , using the transformation described in Section 4.1 for static graphs; recall that the matching output throughout this transformation process is denoted by  $\mathcal{M}^*$ . We may assume that  $|\mathcal{M}|, |\mathcal{M}'| = \Omega(1/\varepsilon)$ , where the constant hiding in the  $\Omega$ -notation is sufficiently large; indeed, otherwise  $|\mathcal{M}| + |\mathcal{M}'| = O(1/\varepsilon)$  and there is no need to apply the transformation process, as the trivial worst-case recourse bound is  $O(1/\varepsilon)$ .

We will show (Lemma 9) that the output matching  $\tilde{\mathcal{M}}_i$  provides a  $(\beta(1 + O(\varepsilon)))$ -MCM at any update step  $i$ . Two simple adjustments are needed for adapting the transformed matching  $\mathcal{M}^*$  of the static setting to the fully dynamic setting:

- To achieve a low worst-case recourse bound and guarantee that the overhead in the update time (with respect to the original update time) is low in the worst-case, we cannot carry out the entire computation at once (i.e. following a single update step), but should rather *simulate it gradually* over the entire time window of the transformation process. Specifically, recall that the transformation process for static graphs consists of two phases, a preprocessing phase in which the matching  $\mathcal{M}' = \mathcal{M}_t^A$  and the sets  $\mathcal{G}$  and  $\mathcal{B}$  of good and bad edges in  $\mathcal{M}' \setminus \mathcal{M}$  are computed, and the actual transformation phase that transforms  $\mathcal{M}^*$ , which is initialized as  $\mathcal{M}$ , into (a possibly superset of)  $\mathcal{M}'$ . Each of these phases requires time  $O(|\mathcal{M}| + |\mathcal{M}'|) = O(|\mathcal{M}|)$ . The first phase does not make any replacements to  $\mathcal{M}^*$ , whereas the second phase consists of a sequence of at most  $|\mathcal{M}'|$  constant-time operations, each of which may trigger a constant number of replacements to  $\mathcal{M}^*$ . The computation of the first phase is simulated in the first  $W/2$  update steps of the window, performing  $O(|\mathcal{M}| + |\mathcal{M}'|)/(W/2) = O(1/\varepsilon)$  computation steps and zero replacements to  $\mathcal{M}^*$  following every update step. The computation of the second phase is simulated in the second  $W/2$  update steps of the window, performing  $O(|\mathcal{M}| + |\mathcal{M}'|)/(W/2) = O(1/\varepsilon)$  computation steps and replacements to  $\mathcal{M}^*$  following every update step.
- Denote by  $\mathcal{M}_i^*$  the matching output at the  $i$ th update step by the resulting gradual transformation process, which simulates  $O(1/\varepsilon)$  computation steps and replacements to the output matching following every update step. While  $\mathcal{M}_i^*$  is a valid matching for the (static) graph  $G_t$  at the beginning of the time window, some of its edges may get deleted from the graph in subsequent update steps  $i = t + 1, t + 2, \dots, t'$ . Consequently, the matching that we shall output for graph  $G_i$ , denoted by  $\tilde{\mathcal{M}}_i$ , is the one obtained from  $\mathcal{M}_i^*$  by removing from it all edges that got deleted from the graph between steps  $t$  and  $i$ .

Once the current time window terminates, a new time window starts, and the same transformation process is repeated, with  $\tilde{\mathcal{M}}_{t'}$  serving as  $\mathcal{M}$  and  $\mathcal{M}_{t'}^A$  serving as  $\mathcal{M}'$ . Since all time windows are handled in the same way, it suffices to analyze the output matching of the current time window, and this analysis would carry over to the entire update sequence.



It is immediate that the output matching  $\tilde{\mathcal{M}}_i$  is a valid matching for any  $i = t, t+1, \dots, t'$ . Moreover, since we make sure to simulate  $O(1/\varepsilon)$  computation steps and replacements following every update step, the worst-case recourse bound of the resulting algorithm is bounded by  $O(1/\varepsilon)$  and the update time is bounded by  $T + O(1/\varepsilon)$ , where this time bound is worst-case/amortized if the time bound  $T$  of Algorithm  $\mathcal{A}$  is worst-case/amortized.

It is left to bound the approximation guarantee of the output matching  $\tilde{\mathcal{M}}_i$ . Recall that  $W = \Theta(\varepsilon \cdot |\mathcal{M}|)$ , and write  $W = \varepsilon' \cdot |\mathcal{M}|$ , with  $\varepsilon' = \Theta(\varepsilon)$ . (We assume that  $\varepsilon$  is sufficiently small so that  $\varepsilon' \leq 1/2$ . We need this restriction on  $\varepsilon'$  to apply Lemma 5.)

► **Lemma 9.**  *$\tilde{\mathcal{M}}_t$  and  $\tilde{\mathcal{M}}_{t'}$  provide a  $(\beta(1+2\varepsilon'))$ -MCM for  $G_t$  and  $G_{t'}$ , respectively. Moreover,  $\tilde{\mathcal{M}}_i$  provides a  $(\beta((1+2\varepsilon')^2))$ -MCM for  $G_i$ , for any  $i = t, t+1, \dots, t'$ .*

**Proof.** First, we bound the approximation guarantee of the matching  $\tilde{\mathcal{M}}_{t'}$ , which is obtained from  $\mathcal{M}_{t'}^*$  by removing from it all edges that got deleted from the graph throughout the time window. By the description of the transformation process,  $\mathcal{M}_{t'}^*$  is a superset of  $\mathcal{M}'$ , hence  $\tilde{\mathcal{M}}_{t'}$  is a superset of the matching obtained from  $\mathcal{M}'$  by removing from it all edges that got deleted throughout the time window. Since  $\mathcal{M}'$  is a  $\beta$ -MCM for  $G_t$ , Lemma 5 implies that  $\tilde{\mathcal{M}}_{t'}$  is a  $(\beta(1+2\varepsilon'))$ -MCM for  $G_{t'}$ . More generally, this argument shows that the matching obtained at the end of any time window is a  $(\beta(1+2\varepsilon'))$ -MCM for the graph at that step.

Next, we argue that the matching obtained at the start of any time window (as described above) is a  $(\beta(1+2\varepsilon'))$ -MCM for the graph at that step. This assertion is trivially true for the first time window, where both the matching and the graph are empty. For any subsequent time window, this assertion follows from the fact that the matching at the start of a new time window is the one obtained at the end of the old time window, for which we have already shown that the required approximation guarantee holds. It follows that  $\tilde{\mathcal{M}}_t = \mathcal{M}$  is a  $(\beta(1+2\varepsilon'))$ -MCM for  $G_t$ .

Finally, we bound the approximation guarantee of the output matching  $\tilde{\mathcal{M}}_i$  in the entire time window. (It suffices to consider the interior of the window.) Lemma 7 implies that  $|\mathcal{M}_i^*| \geq \min\{|\mathcal{M}|, |\mathcal{M}'| - 1\}$ , for any  $i = t, t+1, \dots, t'$ . We argue that  $\mathcal{M}_i^*$  is a  $(\beta(1+2\varepsilon'))$ -MCM for  $G_t$ . If  $|\mathcal{M}_i^*| \geq |\mathcal{M}|$ , then this assertion follows from the fact that  $\mathcal{M}$  provides such an approximation guarantee. We henceforth assume that  $|\mathcal{M}_i^*| \geq |\mathcal{M}'| - 1$ . Recall that  $|\mathcal{M}'| = \Omega(1/\varepsilon) = \Omega(1/\varepsilon')$ , where the constants hiding in the  $\Omega$ -notation are sufficiently large, hence removing a single edge from  $\mathcal{M}'$  cannot hurt the approximation guarantee by more than an additive factor of, say  $\varepsilon'$ , i.e., less than  $\beta(2\varepsilon')$ . Since  $\mathcal{M}'$  provides a  $\beta$ -MCM for  $G_t$ , it follows that  $\mathcal{M}_i^*$  is indeed a  $(\beta(1+2\varepsilon'))$ -MCM for  $G_t$ , which completes the proof of the above assertion. Consequently, Lemma 5 implies that  $\tilde{\mathcal{M}}_i$ , which is obtained from  $\mathcal{M}_i^*$  by removing from it all edges that got deleted from the graph between steps  $t$  and  $i$ , is a  $(\beta((1+2\varepsilon')^2))$ -MCM for  $G_i$ . ◀

#### 4.2.1 A general approximation guarantee

In this section we consider the case of a general approximation parameter  $\beta \geq 1$ . The bound on the approximation guarantee of the output matching provided by Lemma 9, namely  $(\beta((1+2\varepsilon')^2))$ , remains unchanged. Recalling that  $\varepsilon' \leq 1/2$ , it follows that the size of  $\mathcal{M}'$  cannot be larger than that of  $\mathcal{M}$  by more than a factor of  $(\beta((1+2\varepsilon')^2)) \leq 2\beta$ . Consequently, the number of computation steps and replacements performed per update step, namely,  $O(|\mathcal{M}| + |\mathcal{M}'|)/(W/2)$ , is no longer bounded by  $O(1/\varepsilon)$ , but rather by  $O(\beta/\varepsilon)$ . To achieve a bound of  $O(1/\varepsilon)$  for a general  $\beta$ , we shall use a matching  $\mathcal{M}''$  different from  $\mathcal{M}'$ , which includes a possibly small fraction of the edges of  $\mathcal{M}'$ . Recall that we can output  $\ell$  arbitrary edges of the matching  $\mathcal{M}' = \mathcal{M}_t^A$  in time (nearly) linear in  $\ell$ , for any integer



$\ell = 1, 2, \dots, |\mathcal{M}'|$ . Let  $\mathcal{M}''$  be a matching that consists of (up to)  $2|\mathcal{M}|$  arbitrary edges of  $\mathcal{M}'$ ; that is, if  $|\mathcal{M}'| > 2|\mathcal{M}|$ ,  $\mathcal{M}''$  consists of  $2|\mathcal{M}|$  arbitrary edges of  $\mathcal{M}'$ , otherwise  $\mathcal{M}'' = \mathcal{M}'$ . We argue that  $\mathcal{M}''$  is a  $\beta$ -MCM for  $G_t$ . Indeed, if  $|\mathcal{M}'| > 2|\mathcal{M}|$  the approximation guarantee follows from the approximation guarantee of  $\mathcal{M}$  and the fact that  $\mathcal{M}''$  is twice larger than  $\mathcal{M}$ , whereas in the complementary case the approximation guarantee follows from that of  $\mathcal{M}'$ . In any case it is immediate that  $|\mathcal{M}''| = O(|\mathcal{M}|)$ . (For concreteness, we assume that the time needed for storing the edges of  $\mathcal{M}''$  in an appropriate list is  $O(|\mathcal{M}''|) = O(|\mathcal{M}|)$ .) We may henceforth carry out the entire transformation process with  $\mathcal{M}''$  taking the role of  $\mathcal{M}'$ , and in this way guarantee that the number of computation steps and replacements to the output matching performed per update step is reduced from  $O(\beta/\varepsilon)$  to  $O(1/\varepsilon)$ .

### 4.3 Proof of Theorem 4

The proof of Theorem 4 is very similar to the one of Theorem 3. Specifically, we derive Theorem 4 by making a couple of simple adjustments to the proof of Theorem 3 given above, which we sketch next. First, instead of using the transformation of Theorem 1, we use the one of Theorem 2, whose proof appears in Appendix D of the full version [55]. Second, the *stability* property of unweighted matchings used in the proof of Theorem 3 is that the maximum matching size changes by at most 1 following each update step. This stability property enables us in the proof of Theorem 3 to consider a time window of  $W = \Theta(\varepsilon \cdot |\mathcal{M}|)$  update steps, so that any  $\beta$ -MCM computed at the beginning of the window will provide (after removing from it all the edges that get deleted from the graph) a  $(\beta(1 + \varepsilon))$ -MCM throughout the entire window, for any  $\beta \geq 1$ . It is easy to see that this stability property generalizes for weighted matchings, where the maximum matching weight may change by an additive factor of at most  $\psi$ . (Recall that the aspect ratio of the dynamic graph is always bounded by  $\psi$ ; also, we may assume by scaling that the minimum edge weight is 1.) In order to obtain a  $(\beta(1 + \varepsilon))$ -MWM throughout the entire time window, it suffices to consider a time window of  $W' = W/\psi = \Theta(\varepsilon \cdot |\mathcal{M}|/\psi)$ , i.e., a time window shorter than that used for unweighted matchings by a factor of  $\psi$ , and as a result the update time of the resulting algorithm will grow from  $T + O(1/\varepsilon)$  to  $T + O(\psi/\varepsilon)$  and the worst-case recourse bound will grow from  $O(1/\varepsilon)$  to  $O(\psi/\varepsilon)$ .

## 5 Optimality of our Transformations

### 5.1 Unweighted matchings

In the unweighted case, when  $|\mathcal{M}| < |\mathcal{M}'|$ , Theorem 1 states that  $\mathcal{M}$  can gradually transform into  $\mathcal{M}'$  without ever being in deficit compared to the initial value of  $\mathcal{M}$ , i.e.,  $|\mathcal{M}^*| \geq |\mathcal{M}|$  throughout the entire transformation process. If  $|\mathcal{M}'| \leq |\mathcal{M}|$ , however, this no longer holds; in this case the theorem states that we'll reach a deficit of at most 1 unit. To see that this bound is tight, consider the case when  $|\mathcal{M}| = |\mathcal{M}'|$  and  $H = \mathcal{M} \oplus \mathcal{M}'$  is a simple alternating cycle that consists of all edges in  $\mathcal{M}$  and  $\mathcal{M}'$ , and thus of length  $2|\mathcal{M}|$ . Throughout any transformation process and until handling the last edge of the cycle, it must be that  $|\mathcal{M}^*| < |\mathcal{M}|$  if  $\Delta < 2|\mathcal{M}|$ .

► **Remark.** In fact, the same situation will occur if  $\Delta = 2$ . In the particular case of  $\Delta = 1$ , we'll be in deficit of up to 2 throughout the process – adding the first edge of  $\mathcal{M}'$  requires us to delete its two incident edges in  $\mathcal{M}$ , which already leads to a deficit of 2 units.

## 5.2 Weighted matchings

In the weighted case, quantifying this deficit throughout the process is more subtle, but the worst-case scenario remains essentially the same:  $|\mathcal{M}| = |\mathcal{M}'|$ , all edges have weight  $W$  and  $H = \mathcal{M} \oplus \mathcal{M}'$  is a simple alternating cycle that consists of all edges in  $\mathcal{M}$  and  $\mathcal{M}'$ . Throughout any transformation process and until handling the last edge of the cycle, it must be that  $w(\mathcal{M}^*) \leq w(\mathcal{M}) - W$  if  $\Delta < 2|\mathcal{M}|$ . In general, the deficit to the weight of the matching is inverse linear in  $\Delta$ , hence taking  $\Delta$  to be  $\Theta(1/\varepsilon)$  ensures that the weight of the matching throughout the process never goes below  $(1 - \varepsilon)w(\mathcal{M})$ . Interestingly, here a similar situation occurs also when  $w(\mathcal{M}') > w(\mathcal{M})$ . Specifically, consider the same example as above but add  $\eta$  to each edge weight of  $\mathcal{M}'$ , i.e., assume that the edge weights of  $\mathcal{M}$  and  $\mathcal{M}'$  are now  $W$  and  $W' = W + \eta$ , respectively. Then the deficit is no longer  $W$  as before (for  $1 < \Delta < 2|\mathcal{M}|$ ), but rather  $W - \lfloor \Delta/2 \rfloor \cdot \eta$  (for  $1 < \Delta < 2|\mathcal{M}|$ ). Indeed, adding the first edge of  $\mathcal{M}'$  requires the deletion of its two incident edges in  $\mathcal{M}$ , at which stage the deficit is  $W - \eta$ ; from that moment onwards, a single edge of  $\mathcal{M}$  is deleted so that another edge of  $\mathcal{M}'$  can be added, which reduces  $\eta$  from the deficit each time. Therefore, if  $\Delta \cdot \eta = W$ , the deficit is always at least  $W/2$ , while  $w(\mathcal{M}') > w(\mathcal{M}) + W/2$ . This scenario shows that the bound of Theorem 2 is asymptotically tight.

► **Remark.** If  $\Delta = 1$ , we'll be in deficit of  $2W$  (rather than  $W$ ) throughout the process, similarly to the unweighted case. In the degenerate case that  $\mathcal{M}$  and  $\mathcal{M}'$  consist of a single edge each and  $\Delta = 1$ , the weight after the first edge deletion reduces to 0.

## 6 Optimality of the Recourse Bound

In this section we show that an approximation guarantee of  $(1 + \varepsilon)$  requires a recourse bound of  $\Omega(1/\varepsilon)$ , even in the amortized sense and even in the incremental (insertion only) and decremental (deletion only) settings. We only consider edge updates, but the argument extends seamlessly to vertex updates. This lower bound of  $\Omega(1/\varepsilon)$  on the recourse bound does not depend on the update time of the algorithm in any way. Let us fix  $\varepsilon$  to be any parameter satisfying  $\varepsilon = \Omega(1/n)$ ,  $\varepsilon \ll 1$ , where  $n$  is the (fixed) number of vertices.

Consider a simple path  $P_\ell = (v_1, v_2, \dots, v_{2\ell})$  of length  $2\ell - 1$ , for an integer  $\ell = c(1/\varepsilon)$  such that  $\ell \geq 1$  and  $c$  is a sufficiently small constant. (Thus  $P_\ell$  spans at least two but no more than  $n$  vertices.) There is a single maximum matching  $\mathcal{M}_\ell^{OPT}$  for  $P_\ell$ , of size  $\ell$ , which is also the only  $(1 + \varepsilon)$ -MCM for  $P_\ell$ . After adding the two edges  $(v_0, v_1)$  and  $(v_{2\ell}, v_{2\ell+1})$  to  $P_\ell$ , the maximum matching  $\mathcal{M}_\ell^{OPT}$  for the old path  $P_\ell$  does not provide a  $(1 + \varepsilon)$ -MCM for the new path,  $(v_0, v_1, \dots, v_{2\ell+1})$ , which we may rewrite as  $P_{\ell+1} = (v_1, v_2, \dots, v_{2(\ell+1)})$ . The only way to restore a  $(1 + \varepsilon)$ -approximation guarantee is by removing all  $\ell$  edges of  $\mathcal{M}_\ell^{OPT}$  and adding the remaining  $\ell + 1$  edges instead, which yields  $\mathcal{M}_{\ell+1}^{OPT}$ . One may carry out this argument repeatedly until the length of the path reaches, say,  $4\ell - 1$ . The amortized number of replacements to the matching per update step throughout this process is  $\Omega(1/\varepsilon)$ . Moreover, the same amortized bound, up to a small constant factor, holds if we start from an empty path instead of a path of length  $2\ell - 1$ . We then delete all  $4\ell - 1$  edges of the final path and start again from scratch, which may reduce the amortized bound by another small constant. In this way we get an amortized recourse bound of  $\Omega(1/\varepsilon)$  for the fully dynamic setting.

To adapt this lower bound to the incremental setting, we construct  $n' = \Theta(\varepsilon \cdot n)$  vertex-disjoint copies  $P^1, P^2, \dots, P^{n'}$  of the aforementioned incremental path, one after another, in the following way. Consider the  $i$ th copy  $P^i$ , from the moment its length becomes  $2\ell - 1$  and until it reaches  $4\ell - 1$ . If at any moment during this gradual construction of  $P^i$ , the matching

restricted to  $P^i$  is not the (only) maximum matching for  $P^i$ , we *halt* the construction of  $P^i$  and move on to constructing the  $(i + 1)$ th copy  $P^{i+1}$ , and then subsequent copies, in the same way. A copy whose construction started but was halted is called *incomplete*; otherwise it is *complete*. (There are also *empty* copies, whose construction has not started yet.) For any incomplete copy  $P^j$ , the matching restricted to it is not the maximum matching for  $P^j$ , hence its approximation guarantee is worse than  $1 + \varepsilon$ ; more precisely, the approximation guarantee provided by any matching other than the maximum matching for  $P^j$  is at least  $1 + c' \cdot \varepsilon$ , for a constant  $c'$  that can be made as large as we want by decreasing the aforementioned constant  $c$ , or equivalently,  $\ell$ . (Recall that  $\ell = c(1/\varepsilon)$ .) If the matching restricted to  $P^j$  is changed to the maximum matching for  $P^j$  at some later moment in time, we return to that incomplete copy and resume its construction from where we left off, thereby *temporarily suspending* the construction of some other copy  $P_{j'}$ . The construction of  $P^j$  may get halted again, in which case we return to handling the temporarily suspended copy  $P_{j'}$ , otherwise we return to handling  $P_{j'}$  only after the construction of  $P^j$  is complete, and so forth. In this way we maintain the invariant that the approximation guarantee of the matching restricted to any incomplete copy (whose construction is not temporarily suspended) is at least  $1 + c' \cdot \varepsilon$ , for a sufficiently large constant  $c'$ . While incomplete copies may get completed later on, a complete copy remains complete throughout the entire update sequence. At the end of the update sequence no copy is empty or temporarily suspended, i.e., any copy at the end of the update sequence is either incomplete or complete. The above argument implies that any complete copy has an amortized recourse bound of  $\Omega(1/\varepsilon)$ , over the update steps restricted to that copy. Observe also that at least a constant fraction of the  $n'$  copies must be complete at the end of the update sequence, otherwise the entire matching cannot provide a  $(1 + \varepsilon)$ -MCM for the entire graph, i.e., the graph obtained from the union of these  $n'$  copies. It follows that the amortized recourse bound over the entire update sequence is  $\Omega(1/\varepsilon)$ .

The lower bound for the incremental setting can be extended to the decremental setting using a symmetric argument to the one given above.

## 7 Discussion

This paper introduces a natural generalization of the MRP, and provides near-optimal transformations for the problems of MCM and MWM.

One application of this meta-problem is to dynamic graph algorithms. In particular, by building on our transformation for maximum cardinality matching we have shown that any algorithm for maintaining a  $\beta$ -MCM can be transformed into an algorithm for maintaining a  $\beta(1 + \varepsilon)$ -MCM with essentially the same update time as that of the original algorithm and with a worst-case recourse bound of  $O(1/\varepsilon)$ , for any  $\beta \geq 1$  and  $\varepsilon > 0$ . This recourse bound is optimal for the regime  $\beta = 1 + \varepsilon$ . We also extended this result for weighted matchings, but there is a linear dependence on the aspect-ratio of the graph in the update time and recourse bounds. It would be interesting to improve this dependency to be polylogarithmic in the aspect-ratio.

A natural direction for future work is to study additional basic graph problems under this generalized framework. Although our positive results may lead to the impression that there exists an efficient gradual transformation process to any optimization graph problem, we conclude with a sketch of two trivial hardness results.

For the *maximum independent set problem* any gradual transformation process cannot provide any nontrivial approximation guarantee, regardless of the approximation guarantees of the source and target independent sets. To see this, denote the source approximate

maximum independent set (the one we start from) by  $\mathcal{S}$  and the target approximate maximum independent set (the one we gradually transform into) by  $\mathcal{S}'$ , and suppose there is a complete bipartite graph between  $\mathcal{S}$  and  $\mathcal{S}'$ . Since we cannot add even a single vertex of  $\mathcal{S}'$  to the output independent set  $\mathcal{S}^*$  (which is initialized as  $\mathcal{S}$ ) before removing from it all vertices of  $\mathcal{S}$  and assuming each step of the transformation process makes only  $\Delta$  changes to  $\mathcal{S}^*$ , the approximation guarantee of the output independent set must reach  $\Omega(|\mathcal{S}'|/\Delta)$  at some moment throughout the transformation process. In other words, the approximation guarantee may be arbitrarily large.

As another example, an analogous argument shows that for the *minimum vertex cover problem*, any gradual transformation process cannot provide an approximation guarantee better than  $\frac{|\mathcal{C}|+|\mathcal{C}'|}{|\mathcal{C}'|} > 2$ , where  $\mathcal{C}$  and  $\mathcal{C}'$  are the source and target vertex covers, respectively. On the other hand, one can easily see that the approximation guarantee throughout the entire transformation process does not exceed  $\frac{|\mathcal{C}|+|\mathcal{C}'|}{|\mathcal{C}^{OPT}|}$ , where  $\mathcal{C}^{OPT}$  is a minimum vertex cover for the graph, by gradually adding all vertices of the target vertex cover  $\mathcal{C}'$  to the output vertex cover  $\mathcal{C}^*$  (which is initialized as  $\mathcal{C}$ ), and later gradually removing the vertices of  $\mathcal{C}$  from the output vertex cover  $\mathcal{C}^*$ .

These examples demonstrate a basic limitation of our generalized framework, and suggest that further research of this framework is required. One interesting direction for further research is studying the maximum independent set and minimum vertex cover problems for bounded degree graphs; note that the trivial hardness results mentioned above do not apply directly to bounded degree graphs. More generally, studying additional combinatorial optimization problems under this framework may contribute to a deeper understanding of its inherent limitations and strengths, and in particular, to finding additional applications of this framework, possibly outside the area of dynamic matching algorithms.

---

## References

- 1 Ittai Abraham, David Durfee, Ioannis Koutis, Sebastian Krinninger, and Richard Peng. On fully dynamic graph sparsifiers. In *Proc. of 57th FOCS*, pages 335–344, 2016.
- 2 Spyros Angelopoulos, Christoph Dürr, and Shendan Jin. Online maximum matching with recourse. In *Proc. of 43rd MFCS*, pages 8:1–8:15, 2018.
- 3 Moab Arar, Shiri Chechik, Sarel Cohen, Cliff Stein, and David Wajc. Dynamic matching: Reducing integral algorithms to approximately-maximal fractional algorithms. In *Proc. 45th ICALP*, pages 7:1–7:16, 2018.
- 4 Sepehr Assadi, MohammadHossein Bateni, Aaron Bernstein, Vahab S. Mirrokni, and Cliff Stein. Coresets meet EDCS: algorithms for matching and vertex cover on massive graphs. *CoRR*, abs/1711.03076, 2017. [arXiv:1711.03076](https://arxiv.org/abs/1711.03076).
- 5 Sepehr Assadi and Sanjeev Khanna. Randomized composable coresets for matching and vertex cover. In *Proc. of 29th SPAA*, pages 3–12, 2017.
- 6 Sepehr Assadi, Krzysztof Onak, Baruch Schieber, and Shay Solomon. Fully dynamic maximal independent set with sublinear update time. In *Proc. 50th STOC*, 2018.
- 7 Giorgio Ausiello, Vincenzo Bonifaci, and Bruno Escoffier. Complexity and approximation in reoptimization. In *Computability in Context: Computation and Logic in the Real World*, pages 101–129. World Scientific, 2011.
- 8 Giorgio Ausiello, Bruno Escoffier, Jérôme Monnot, and Vangelis Th. Paschos. Reoptimization of minimum and maximum traveling salesman’s tours. *J. Discrete Algorithms*, 7(4):453–463, 2009.
- 9 Nikhil Bansal, Anupam Gupta, Ravishankar Krishnaswamy, Kirk Pruhs, Kevin Schewior, and Clifford Stein. A 2-competitive algorithm for online convex optimization with switching costs. In *Proc. of APPROX-RANDOM*, pages 96–109, 2015.
- 10 S. Baswana, M. Gupta, and S. Sen. Fully dynamic maximal matching in  $O(\log n)$  update time. In *Proc. of 52nd FOCS*, pages 383–392, 2011.

- 11 Michael A Bender, Martin Farach-Colton, Sándor P Fekete, Jeremy T Fineman, and Seth Gilbert. Reallocation problems in scheduling. *Algorithmica*, 73(2):389–409, 2015.
- 12 Michael A Bender, Martin Farach-Colton, Sándor P Fekete, Jeremy T Fineman, and Seth Gilbert. Cost-oblivious storage reallocation. *TALG*, 13(3):38, 2017.
- 13 Aaron Bernstein, Sebastian Forster, and Monika Henzinger. A deamortization approach for dynamic spanner and dynamic maximal matching. In *Proc. SODA*, pages 1899–1918, 2019.
- 14 Aaron Bernstein, Jacob Holm, and Eva Rotenberg. Online bipartite matching with amortized  $O(\log^2 n)$  replacements. In *Proc. of 28th SODA*, pages 692–711, 2018.
- 15 Aaron Bernstein, Tsvi Kopelowitz, Seth Pettie, Ely Porat, and Clifford Stein. Simultaneously load balancing for every p-norm, with reassignments. In *Proc. ITCS*, pages 51:1–51:14, 2017.
- 16 Aaron Bernstein and Cliff Stein. Fully dynamic matching in bipartite graphs. In *Proc. 42nd ICALP*, pages 167–179, 2015.
- 17 Aaron Bernstein and Cliff Stein. Faster fully dynamic matchings with small approximation ratios. In *Proc. of 26th SODA*, pages 692–711, 2016.
- 18 S. Bhattacharya, M. Henzinger, and D. Nanongkai. Fully dynamic maximum matching and vertex cover in  $o(\log^3 n)$  worst case update time. In *Proc. of 28th SODA*, pages 470–489, 2017.
- 19 Davide Bilò. New algorithms for steiner tree reoptimization. In *Proc. of 45th ICALP*, pages 19:1–19:14, 2018. doi:10.4230/LIPIcs.ICALP.2018.19.
- 20 Davide Bilò, Hans-Joachim Böckenhauer, Dennis Komm, Richard Královic, Tobias Mömke, Sebastian Seibert, and Anna Zych. Reoptimization of the shortest common superstring problem. *Algorithmica*, 61(2):227–251, 2011.
- 21 Marthe Bonamy, Nicolas Bousquet, Marc Heinrich, Takehiro Ito, Yusuke Kobayashi, Arnaud Mary, Moritz Mühlenthaler, and Kunihiro Wasa. The perfect matching reconfiguration problem. In *Proc. of 44th MFCS*, volume 138 of *LIPIcs*, pages 80:1–80:14, 2019.
- 22 Paul Bonsma and Luis Cereceda. Finding paths between graph colourings: Pspace-completeness and superpolynomial distances. *Theoretical Computer Science*, 410(50):5215–5226, 2009.
- 23 Nicolas Boria and Vangelis Th. Paschos. Fast reoptimization for the minimum spanning tree problem. *J. Discrete Algorithms*, 8(3):296–310, 2010.
- 24 Bartłomiej Bosek, Dariusz Leniowski, Piotr Sankowski, and Anna Zych. Online bipartite matching in offline time. In *Proc. 55th FOCS*, pages 384–393, 2014.
- 25 Bartłomiej Bosek, Dariusz Leniowski, Piotr Sankowski, and Anna Zych. Shortest augmenting paths for online matchings on trees. In *Proc. of 13th WAOA*, pages 59–71, 2015.
- 26 Bartłomiej Bosek, Dariusz Leniowski, Piotr Sankowski, and Anna Zych-Pawlewicz. A tight bound for shortest augmenting paths on trees. In *Proc. 13th LATIN*, pages 201–216, 2018.
- 27 Nicolas Bousquet, Tatsuhiko Hatanaka, Takehiro Ito, and Moritz Mühlenthaler. Shortest reconfiguration of matchings. In *Proc. of 45th WG*, pages 162–174. Springer, 2019.
- 28 Keren Censor-Hillel, Elad Haramaty, and Zohar S. Karnin. Optimal dynamic distributed MIS. In *Proc. of PODC*, pages 217–226, 2016.
- 29 Moses Charikar and Shay Solomon. Fully dynamic almost-maximal matching: Breaking the polynomial worst-case time barrier. In *Proc. 45th ICALP*, pages 33:1–33:14, 2018.
- 30 Kamalika Chaudhuri, Constantinos Daskalakis, Robert D. Kleinberg, and Henry Lin. Online bipartite perfect matching with augmentations. In *Proc. INFOCOM*, pages 1044–1052, 2009.
- 31 P. Gopalan, P. G Kolaitis, E. Maneva, and C. H Papadimitriou. The connectivity of boolean satisfiability: computational and structural dichotomies. *SICOMP*, 38(6):2330–2355, 2009.
- 32 Fabrizio Grandoni, Stefano Leonardi, Piotr Sankowski, Chris Schwegelshohn, and Shay Solomon.  $(1 + \epsilon)$ -approximate incremental matching in constant deterministic amortized time. In *Proc. of 30th SODA*, pages 1886–1898, 2019.
- 33 Edward F. Grove, Ming-Yang Kao, P. Krishnan, and Jeffrey Scott Vitter. Online perfect matching and mobile computing. In *Proc. of 45th Wads*, pages 194–205, 1995.
- 34 Albert Gu, Anupam Gupta, and Amit Kumar. The power of deferral: maintaining a constant-competitive steiner tree online. In *Proc. of STOC*, pages 525–534, 2013.
- 35 Anupam Gupta, Ravishankar Krishnaswamy, Amit Kumar, and Debmalya Panigrahi. Online and dynamic algorithms for set cover. In *Proc. 49th STOC*, pages 537–550, 2017.


- 36 Anupam Gupta and Amit Kumar. Online steiner tree with deletions. In *Proc. of 25th SODA*, pages 455–467, 2014.
- 37 Anupam Gupta, Amit Kumar, and Cliff Stein. Maintaining assignments online: Matching, scheduling, and flows. In *Proc. 25th SODA*, pages 468–479, 2014.
- 38 M. Gupta and R. Peng. Fully dynamic  $(1 + \epsilon)$ -approximate matchings. In *54th FOCS*, pages 548–557, 2013.
- 39 Manoj Gupta, Hitesh Kumar, and Neeldhara Misra. On the complexity of optimal matching reconfiguration. In *International Conference on Current Trends in Theory and Practice of Informatics*, pages 221–233. Springer, 2019.
- 40 Robert A. Hearn and Erik D. Demaine. The nondeterministic constraint logic model of computation: Reductions and applications. In *Proc. ICALP*, pages 401–413. Springer, 2002.
- 41 T. Ito, E. D. Demaine, N. J. A. Harvey, C. H. Papadimitriou, M. Sideri, R. Uehara, and Y. Uno. On the complexity of reconfiguration problems. *TCS*, 412(12-14):1054–1065, 2011.
- 42 T. Ito, N. Kakimura, N. Kamiyama, Y. Kobayashi, and Y. Okamoto. Reconfiguration of maximum-weight b-matchings in a graph. *J. Comb. Optim.*, 37(2):454–464, 2019.
- 43 Takehiro Ito, Naonori Kakimura, Naoyuki Kamiyama, Yusuke Kobayashi, and Yoshio Okamoto. Shortest reconfiguration of perfect matchings via alternating cycles. In *Proc. of 27th ESA*, volume 144 of *LIPICs*, pages 61:1–61:15, 2019.
- 44 Marcin Kaminski, Paul Medvedev, and Martin Milanic. Complexity of independent set reconfigurability problems. *Theor. Comput. Sci.*, 439:9–15, 2012.
- 45 Bruce M. Kapron, Valerie King, and Ben Mountjoy. Dynamic graph connectivity in polylogarithmic worst case time. In *Proc. of 24th SODA*, pages 1131–1142, 2013.
- 46 Subhash Khot and Oded Regev. Vertex cover might be hard to approximate to within 2-epsilon. *J. Comput. Syst. Sci.*, 74(3):335–349, 2008.
- 47 Jannik Matuschke, Ulrike Schmidt-Kraepelin, and José Verschae. Maintaining perfect matchings at low cost. *CoRR*, abs/1811.10580, 2018. [arXiv:1811.10580](#).
- 48 Nicole Megow, Martin Skutella, José Verschae, and Andreas Wiese. The power of recourse for online MST and TSP. *SIAM J. Comput.*, 45(3):859–880, 2016.
- 49 Vahab S. Mirrokni and Morteza Zadimoghaddam. Randomized composable core-sets for distributed submodular maximization. In *Proc. 47th STOC*, pages 153–162, 2015.
- 50 Ofer Neiman and Shay Solomon. Simple deterministic algorithms for fully dynamic maximal matching. In *Proc. 45th STOC*, pages 745–754, 2013.
- 51 Naomi Nishimura. Introduction to reconfiguration. *Algorithms*, 11(4):52, 2018.
- 52 K. Onak and R. Rubinfeld. Maintaining a large matching and a small vertex cover. In *Proc. of 42nd STOC*, pages 457–464, 2010.
- 53 D. Peleg and S. Solomon. Dynamic  $(1 + \epsilon)$ -approximate matchings: A density-sensitive approach. In *Proc. of 26th SODA*, 2016.
- 54 Baruch Schieber, Hadas Shachnai, Gal Tamir, and Tami Tamir. A theory and algorithms for combinatorial reoptimization. *Algorithmica*, 80(2):576–607, 2018.
- 55 Noam Solomon and Shay Solomon. A generalized matching reconfiguration problem. *CoRR*, abs/1803.05825, 2020. [arXiv:1803.05825](#).
- 56 S. Solomon. Fully dynamic maximal matching in constant update time. In *Proc. 57th FOCS*, pages 325–334, 2016.
- 57 Babacar Thiongane, Anass Nagih, and Gérard Plateau. Lagrangean heuristics combined with reoptimization for the 0–1 bidimensional knapsack problem. *Discrete applied mathematics*, 154(15):2200–2211, 2006.
- 58 Jan van den Heuvel. The complexity of change. In Simon R. Blackburn, Stefanie Gerke, and Mark Wildon, editors, *Surveys in Combinatorics 2013*, volume 409 of *London Mathematical Society Lecture Note Series*, pages 127–160. Cambridge University Press, 2013.
- 59 David Wajc. Rounding dynamic matchings against an adaptive adversary. In *Proc. of 52nd STOC*, pages 194–207. ACM, 2020.



# Sensitivity Analysis of the Maximum Matching Problem

Yuichi Yoshida 

National Institute of Informatics, Tokyo, Japan  
yyoshida@nii.ac.jp

Samson Zhou 

Carnegie Mellon University, Pittsburgh, PA, USA  
samsonzhou@gmail.com

---

## Abstract

We consider the *sensitivity* of algorithms for the maximum matching problem against edge and vertex modifications. When an algorithm  $A$  for the maximum matching problem is deterministic, the sensitivity of  $A$  on  $G$  is defined as  $\max_{e \in E(G)} |A(G) \Delta A(G - e)|$ , where  $G - e$  is the graph obtained from  $G$  by removing an edge  $e \in E(G)$  and  $\Delta$  denotes the symmetric difference. When  $A$  is randomized, the sensitivity is defined as  $\max_{e \in E(G)} d_{\text{EM}}(A(G), A(G - e))$ , where  $d_{\text{EM}}(\cdot, \cdot)$  denotes the earth mover's distance between two distributions. Thus the sensitivity measures the difference between the output of an algorithm after the input is slightly perturbed. Algorithms with low sensitivity, or *stable* algorithms are desirable because they are robust to edge failure or attack.

In this work, we show a randomized  $(1 - \epsilon)$ -approximation algorithm with *worst-case* sensitivity  $O_\epsilon(1)$ , which substantially improves upon the  $(1 - \epsilon)$ -approximation algorithm of Varma and Yoshida (SODA'21) that obtains *average* sensitivity  $n^{O(1/(1+\epsilon^2))}$  sensitivity algorithm, and show a deterministic  $1/2$ -approximation algorithm with sensitivity  $\exp(O(\log^* n))$  for bounded-degree graphs. We then show that any deterministic constant-factor approximation algorithm must have sensitivity  $\Omega(\log^* n)$ . Our results imply that randomized algorithms are strictly more powerful than deterministic ones in that the former can achieve sensitivity independent of  $n$  whereas the latter cannot. We also show analogous results for vertex sensitivity, where we remove a vertex instead of an edge.

Finally, we introduce the notion of normalized weighted sensitivity, a natural generalization of sensitivity that accounts for the weights of deleted edges. For a graph with weight function  $w$ , the normalized weighted sensitivity is defined to be the sum of the weighted edges in the symmetric difference of the algorithm normalized by the altered edge, i.e.,  $\max_{e \in E(G)} \frac{1}{w(e)} w(A(G) \Delta A(G - e))$ . Hence the normalized weighted sensitivity measures the weighted difference between the output of an algorithm after the input is slightly perturbed, normalized by the weight of the perturbation. We show that if all edges in a graph have polynomially bounded weight, then given a trade-off parameter  $\alpha > 2$ , there exists an algorithm that outputs a  $\frac{1}{4\alpha}$ -approximation to the maximum weighted matching in  $O(m \log_\alpha n)$  time, with normalized weighted sensitivity  $O(1)$ .

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Graph algorithms analysis; Theory of computation  $\rightarrow$  Approximation algorithms analysis; Theory of computation  $\rightarrow$  Models of computation

**Keywords and phrases** Sensitivity analysis, maximum matching, graph algorithms

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.58

**Related Version** <https://arxiv.org/pdf/2009.04556.pdf>

**Funding** *Yuichi Yoshida*: Supported by JST, PRESTO Grant Number JPMJPR192B, Japan.  
*Samson Zhou*: Supported by a Simons Investigator Award of David P. Woodruff.

**Acknowledgements** We would like to thank anonymous ITCS reviewers for helpful comments on an earlier version of the paper.



© Yuichi Yoshida and Samson Zhou;

licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 58; pp. 58:1–58:20

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



## 1 Introduction

The problem of finding the maximum matching in a graph is a fundamental problem in graph theory with a wide range of applications in computer science. For example, the maximum matching problem on a bipartite graph  $G$  captures a typical example where a number of possible clients want to access content distributed across multiple providers. Each client can download their specific content from a specific subset of the possible providers, but each provider can only connect to a limited number of clients. A maximum matching between clients and providers would ensure that the largest possible number of clients receive their content.

However in many modern applications, the underlying graph  $G$  represents some large dataset that is often dynamic or incomplete. In the above example, the content preference of clients may change, which alters the set of suppliers that provide their desired content. Connections between specific providers and clients may become online or offline, effectively adding or removing edges in the underlying graph. Providers and clients may themselves join or leave the network, adding or removing entire vertices from the graph. Thus, it is reasonable to assume that our knowledge of important properties of  $G$  may also change or be incomplete. Nevertheless, we must extract information from our current knowledge of  $G$  either for pre-processing or to perform tasks on the current infrastructure. At the same time, we would like to maintain as much consistency as possible when updates to  $G$  are revealed.

Motivated by a formal definition of consistency of algorithms across graph updates, Varma and Yoshida [12] first defined the *average sensitivity* of a deterministic algorithm  $A$  to be the Hamming distance<sup>1</sup> between the output of  $A$  on graphs  $G$  and  $G - e$ , where  $G'$  is the graph formed by deleting a random edge of  $G$ . Then, they defined *average sensitivity* for randomized algorithms as

$$\mathbb{E}_{e \sim E(G)} [d_{\text{EM}}(A(G), A(G - e))],$$

where  $d_{\text{EM}}(\cdot, \cdot)$  denotes the earth mover's distance and  $G - e$  is the graph obtained from  $G$  by deleting an edge  $e \in E(G)$ . For the maximum matching problem, they showed a randomized  $1/2$ -approximation algorithm with average sensitivity  $O(1)$  and a randomized  $(1 - \epsilon)$ -approximation algorithm with average sensitivity  $O(n^{1/(1+\epsilon^2)})$ .

**Worst case sensitivity.** In this work, we continue the study of sensitivity for the maximum matching problem. Instead of average sensitivity as in [12], we consider a stronger notion of (worst-case) sensitivity. Specifically, the *sensitivity* of a deterministic algorithm  $A$  is the maximum Hamming distance between the output of  $A$  on graphs  $G$  and  $G'$ , where  $G'$  is the graph formed by deleting an edge of  $G$ . Then, the *sensitivity* of a randomized algorithm  $A$  is

$$\max_{e \in E(G)} d_{\text{EM}}(A(G), A(G - e)).$$

Clearly, the sensitivity of an algorithm is no smaller than its average sensitivity. As a natural variant, we also consider *vertex sensitivity*, where we delete a vertex instead of an edge. To avoid confusion, sensitivity with respect edge deletion will be sometimes called *edge sensitivity*.

---

<sup>1</sup> Here we regard the output as a binary string so we can think of the Hamming distance between outputs.

## 1.1 Our Contributions

We first show that, for any  $\epsilon > 0$ , there exists a randomized  $(1 - \epsilon)$ -approximation algorithm whose sensitivity solely depends on  $\epsilon$  (Section 2).

► **Theorem 1.** *For any  $\epsilon > 0$ , there exists an algorithm that outputs a  $(1 - \epsilon)$ -approximation to the maximum matching problem with probability at least 0.99, using time complexity  $O((n + m) \cdot K)$  and edge/vertex sensitivity  $O(3^K)$ , where  $K = (1/\epsilon)^{2^{O(1/\epsilon)}}$ .*

This result improves upon the previous  $(1 - \epsilon)$ -approximation algorithm [12] in that (1) the sensitivity is constant instead of  $O(n^{1/(1+\epsilon^2)})$  and (2) it bounds worst-case sensitivity instead of average sensitivity.

We observe that approximation is necessary to achieve a small sensitivity. For example, consider an  $n$ -cycle for an even  $n$ , and let  $M_1$  and  $M_2$  be the two maximum matchings of size  $n/2$  in the graph. Consider a deterministic algorithm that always outputs a maximum matching, say,  $M_1$  for the  $n$ -cycle. Then, it must output  $M_2$  after removing an edge in  $M_1$ , and hence the sensitivity is  $\Omega(n)$ . With a similar reasoning, we can show a lower bound of  $\Omega(n)$  for randomized algorithms. Also as we show in Section 4.3, the dependency on  $\epsilon$  in Theorem 1 is necessary.

Next, we show a deterministic algorithm for finding a maximal matching on bounded degree graphs that has low sensitivity (Section 3). Note that it has approximation ratio  $1/2$  because the size of any maximal matching is a  $1/2$ -approximation to the maximum matching.

► **Theorem 2.** *There exists a deterministic algorithm that finds a maximal matching with edge/vertex sensitivity  $\Delta^{O(6^\Delta + \log^* n)}$ , where  $\Delta$  is the maximum degree of a vertex in the graph.*

Then, we show that randomness is necessary to achieve sensitivity independent of  $n$  (Section 4):

► **Theorem 3.** *Any deterministic constant-factor approximation algorithm for the maximum matching problem has edge sensitivity  $\Omega(\log^* n)$ .*

Namely, we show in Section 4.2 that we cannot obtain sublinear sensitivity just by derandomizing the randomized greedy algorithm. Theorems 1 and 3 imply that randomized algorithms are strictly more powerful than deterministic ones in that the former can achieve sensitivity independent of  $n$  whereas the latter cannot for the maximum matching problem.

We then introduce the idea of *weighted sensitivity*, which is a natural generalization of sensitivity for both the average and worst cases. For the problems that we consider, the sensitivity of a deterministic graph algorithm is the number of edges that changes in the output induced by the alteration of a single vertex/edge in the input. Thus for a weighted graph, the weighted sensitivity is the total weight of the edges that are changed in the output, following the deletion of a vertex/edge. For randomized algorithms, the definition extends naturally to the earth mover's distance between the distributions with the corresponding weighted loss function. Finally, we can also normalize by the weight of the edge that is deleted.

The motivation for studying weighted sensitivity is natural; in many applications with evolving data, the notion of sensitivity arises in the context of recourse, a quantity that measures the change in the underlying topology of the optimal solution. For example in the facility location problem, the goal is to construct a set of facilities to minimize the sum of

the costs of construction and service to a set of consumers. As the information about the set of consumers evolves, it would be ideal to minimize the number of relocations for the facilities, due to the construction costs, which is measured by the sensitivity of the algorithm. However, as construction costs may not be uniform, a more appropriate quantity to minimize would be the total cost of the relocations for the facilities, which is measured by the weighted sensitivity.

Similarly, matchings are often used to maximize flow across a bipartite graph, but the physical structures that support the flow may incur varying costs to construct or demolish, corresponding to the amount of flow that the structures support. In this case, we note that it may not be possible for the worst case weighted sensitivity to be small. For example, if a single edge has weight  $n^C$  for some large constant  $C$  and the remaining edges have weight 1, any constant factor approximation to the maximum weighted matching must include the heavy edge. But if the heavy edge is then removed from the graph, the weighted sensitivity of any constant factor approximation algorithm is  $\Omega(n^C)$ . This issue is circumvented by the normalized weighted sensitivity, which scales the sensitivity by the weight of the deleted edge. We give approximation algorithms for maximum weighted matching with low normalized weighted worst-case sensitivity.

► **Theorem 4.** *Let  $G = (V, E)$  be a weighted graph with  $\frac{1}{n^c} \leq w(e) \leq n^c$  for some constant  $c > 0$  and all  $e \in E$ . For a trade-off parameter  $\alpha > 2$ , there exists an algorithm that outputs a  $\frac{1}{4\alpha}$ -approximation to the maximum weighted matching in  $O(m \log_\alpha n)$  time and has normalized weighted sensitivity  $O(1)$ .*

Our results also extend to  $\alpha = 2$  and general worst-case weighted sensitivity, i.e., weighted sensitivity that is not normalized. We detail these algorithms in Section 5.

## 1.2 Proof Sketch

We explain the idea behind the algorithm of Theorem 1. For simplicity, we focus on edge sensitivity. We note that if we only sought a  $1/2$ -approximation to the maximum matching, then it would suffice to find any maximal matching. Although the well-known greedy algorithm produces a maximal matching, the output of the algorithm is highly sensitive to the ordering of the edges in the input. One may hope that, if we choose an ordering of the edges uniformly at random, then the resulting output will be stable against edge deletions to the underlying graph. This is not immediately obvious because the deleted edge will appear about halfway through the ordering (of the edges in the original graph) in expectation, so it seems possible that it can impact about the remaining half of the edges. Luckily, we show that the edges at the beginning of the ordering are significantly more important, so that even if the deleted edge appears about halfway through the ordering, the sensitivity of the maximal matching is  $O(1)$  (Section 2.3.1). Our analysis is similar to [2], who show that the vertices at the beginning of an ordering are significantly more important in maintaining a maximal independent set in the dynamic distributed model.

Adapting this idea to a  $(1 - \epsilon)$ -approximation is more challenging. The natural approach is to take a maximal matching and repeatedly find a large number of augmenting paths, but the change of even a single edge in a maximal matching can potentially impact a large number of edges if the augmenting paths are found in a sequential manner. We instead adapt a layered graph of [9] that is used to randomly find a large number of augmenting paths in a small number of passes in the streaming model. Crucially, we instead find a large number of disjoint augmenting paths in a small number of parallel rounds, which results in low sensitivity.

Now we turn to explaining the idea behind Theorem 2. Again we focus on edge sensitivity. Our algorithm first uses a deterministic local computation algorithm (LCA) of [4] for  $6^\Delta$ -coloring a graph  $G$  with maximum degree  $\Delta$ , using  $O(\Delta \log^* n)$  probes to an adjacency list oracle. Here we want to design an algorithm that answer queries about the colors of vertices by making a series of probes to the oracle. The answers of the algorithm must be consistent so that there exists at least one proper coloring that is consistent with the answers. In our case, each probe to the oracle is a query  $(v, i)$  with  $v \in V$  and a positive integer  $i$ . If the degree of  $v$  is at least  $i$ , the oracle responds with the  $i$ -th neighbor of  $v$  to the probe. Otherwise, the oracle outputs a special symbol  $\perp$ . In particular, the deterministic  $6^\Delta$ -coloring LCA only probes vertices that are within a “small” neighborhood of the query.

Given a coloring for  $G$ , we then give a local distributed algorithm that takes a coloring of a graph and outputs a maximal matching. It follows from a framework of [10] that our local distributed algorithm can actually be simulated by a deterministic LCA that again only probes a “small” neighborhood of the query. Thus to bound the sensitivity of the algorithm, we bound the number of queries for which a deleted edge would be probed. Since only a small number of queries probes the deleted edge, then the output of the algorithm only has a small number of changes and thus low worst-case sensitivity.

Our lower bound of Theorem 3 considers the set of length- $t$  cycles on a graph with  $n$  vertices. Any matching on length- $t$  cycles can be represented as a series of indicator variables denoting whether edge  $i \in [t]$  is in the matching. We can then interpret the indicator variables as an integer encoding from 0 to  $2^t - 1$  through the natural binary representation. Ramsey theory claims that for  $t = O(\log^* n)$ , there exists a set  $S$  of  $t + 1$  nodes of  $n$  so that any subset of  $t$  nodes has the same encoding. We then choose  $G$  and  $G'$  to be the cycle graphs consisting of the first  $t$  nodes of  $S$  and the last  $t$  nodes of  $S$ , respectively. Since the encodings of the matchings of  $G$  and  $G'$  are the same, but the edge indices are shifted by one, it follows that  $\Omega(t)$  edges must be in the symmetric difference between  $G$  and  $G'$ , which implies from  $t = O(\log^* n)$  that the worst-case sensitivity of the algorithm must be  $\Omega(\log^* n)$ .

### 1.3 Related Work

Varma and Yoshida [12] introduced the notion of sensitivity and performed a systematic study of average sensitivity on many graph problems. Namely, they gave efficient approximation algorithms with low average sensitivities for the minimum spanning forest problem, the global minimum cut problem, the minimum  $s$ - $t$  cut problem, and the maximum matching problem. They also introduced a low-sensitivity algorithm for linear programming, and proved many fundamental properties of average sensitivity, such as sequential or parallel composition. Peng and Yoshida [11] gave an algorithm for the problem of spectral clustering with average sensitivity  $\frac{\lambda_2}{\lambda_3}$ , where  $\lambda_i$  is the  $i$ -th smallest eigenvalue of the normalized Laplacian, which is small when there are exactly two clusters in the graph.

The effects of graph updates have also been studied significantly in the dynamic/online model, where updates to the graph arrive in a stream, and the goal is to maintain some data structure to answer queries on the underlying graph so that both the update time and query time are efficient. Consequently, most of the literature for dynamic algorithms focuses on optimizing these quantities, rather than the changes in the output as the data evolves. Sensitivity analysis is more relevant when the goal of the dynamic/offline model is to minimize the number of changes between successive outputs of the algorithm over the stream.

Lattanzi and Vassilvitski [8] studied the problem of consistent  $k$ -clustering, where the goal is to maintain a constant-factor approximation to some underlying  $k$ -clustering problem, such as  $k$ -center,  $k$ -median, or  $k$ -means, while minimizing the total number of changes to the

set of centers as the stream evolves. In this setting, each change to the set of center is known as a *recourse*. Whereas the model of [8] allows only insertions of new points, algorithms with low sensitivity are robust against both insertions and deletions. Cohen-Addad *et. al.* [3] further considered the facility location problem in this model of maintaining a constant-factor approximation while minimizing the total recourse. Although the algorithm of [3] addresses both the insertions and deletions of points, their total recourse across the stream is  $O(n)$ , where  $n$  is the length of the stream; this is inherent to the difficulty of their problem in the model. Whereas their work already provides an amortized  $O(1)$  recourse per update, we also study the worst-case sensitivity in our work.

## 1.4 Preliminaries

For a positive integer  $n$ , let  $[n]$  denote the set  $\{1, 2, \dots, n\}$ . For a positive integer  $n$  and  $p \in [0, 1]$ , let  $\mathcal{B}(n, p)$  be the binomial distribution with  $n$  trials and success probability  $p$ . We use the notation  $O_\epsilon(\cdot)$  to omit dependencies on  $\epsilon$ .

Let  $G = (V, E)$  be a graph. For an edge  $e \in E$ , let  $N_G(e)$  be the “neighboring” edges of  $e$  in  $G$ , that is,  $N_G(e) = \{e' \in E \mid e' \neq e, |e' \cap e| \geq 1\}$ . We omit the subscript if it is clear from the context.

For two (vertex or edge) sets  $S$  and  $S'$ , let  $d_H(S, S') = |S \Delta S'|$ , where  $\Delta$  denotes the symmetric difference. Abusing the notation, for set of paths  $\mathcal{P}$  and  $\mathcal{P}'$ , we write  $d_H(\mathcal{P}, \mathcal{P}')$  to denote  $d_H(\cup_{P \in \mathcal{P}} V(P), \cup_{P \in \mathcal{P}'} V(P))$ . For two random sets  $X$  and  $X'$ , let  $d_{EM}(X, X')$  be the earth mover’s distance between  $X$  and  $X'$ , where the distance between two sets is measured by  $d_H$ , that is,

$$d_{EM}(X, X') = \min_{\mathcal{D}} \mathbf{E}_{(S, S') \sim \mathcal{D}} d_H(S, S'),$$

where  $\mathcal{D}$  is a distribution such that its marginal on the first and second coordinates are  $X$  and  $X'$ , respectively. For a real-valued function  $\beta$  on graphs, we say that the *sensitivity* of a (randomized) algorithm  $A$  that outputs a set of edges is at most  $\beta$  if

$$d_{EM}(A(G), A(G - e)) \leq \beta(G)$$

holds for every  $e \in E(G)$ .

Given a matching  $M$  in a graph  $G = (V, E)$ , we call a vertex *free* if it does not appear as the endpoint of any edge in  $M$ . A path  $(v_1, v_2, \dots, v_{2\ell+2})$  of length  $2\ell + 1$  is an *augmenting path* if  $v_1$  and  $v_{2\ell+2}$  are free vertices and  $(v_i, v_{i+1}) \in M$  for even  $i$  and  $(v_i, v_{i+1}) \in E \setminus M$  for odd  $i$ .

## 2 Randomized $(1 - \epsilon)$ -Approximation

In this section, we prove Theorem 1. Our algorithm, which we describe in Section 2.1, is a slight modification of the multi-pass streaming algorithm due to McGregor [9]. We discuss its approximation guarantee and sensitivity in Sections 2.2 and 2.3, respectively.

### 2.1 Algorithm Description

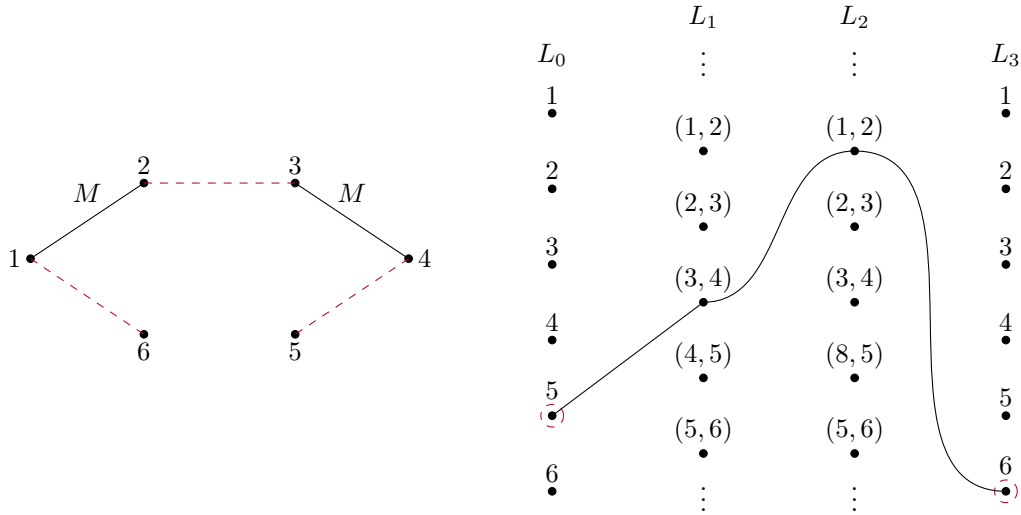
A key step of McGregor’s algorithm is to find a large set of augmenting paths of a specified length in a batch manner using the *layered graph*, given below. Given a graph  $G = (V, E)$ , a matching  $M \subseteq E$ , and a positive integer  $\ell$ , the layered graph  $H = H(G)$  consists of  $\ell + 2$  layers  $L_0, L_1, \dots, L_{\ell+1}$ , where  $L_0 = L_{\ell+1} = V$  and  $L_1 = L_2 = \dots = L_\ell = V \times V$ .

For each vertex  $v \in V$ , we sample  $i_v \in \{0, \ell + 1\}$  uniformly at random independently from others. We say that the copy of  $v$  in the  $L_{i_v}$ -th layer is *active* and that the other copy is *inactive*. For each edge  $\{u, v\} \in M$ , with probability half, we sample a value  $i_{(u,v)} \in \{1, \dots, \ell\}$  uniformly at random and set  $i_{(v,u)} = \perp$ , where  $\perp$  is a special symbol, and with the remaining probability half, we sample a value  $i_{(v,u)} \in \{1, \dots, \ell\}$  uniformly at random and set  $i_{(u,v)} = \perp$ . For each edge  $\{u, v\} \in E \setminus M$ , we set  $i_{(u,v)} = i_{(v,u)} = \perp$ . We say that the copy of  $(u, v)$  in the  $L_{i_{(u,v)}}$ -th is *active* if  $i_{(u,v)} \neq \perp$  and is *inactive* otherwise. Intuitively, some orientation of each edge  $\{u, v\}$  in the matching  $M$  is assigned to a random internal layer in  $H$  and edges of  $G$  that are not in the matching are not initially assigned to any layer in  $H$ . For  $i = 0, \dots, \ell + 1$ , we denote by  $\tilde{L}_i$  the set of active vertices in  $L_i$ . Let  $L = \bigcup_{i=0}^{\ell+1} L_i$  be the vertex set of  $H$ , and let  $\tilde{L} = \bigcup_{i=0}^{\ell+1} \tilde{L}_i$  be the set of active vertices in  $H$ .

The edges in the layered graph  $H$  are those between active vertices that can be a part of an augmenting path in  $G$ . More specifically,

- We add an edge between  $t \in \tilde{L}_0$  and  $(u, v) \in \tilde{L}_1$  if  $t$  is free in  $M$  and  $t$  is adjacent to  $v$ .
- We add an edge between  $(u, v) \in \tilde{L}_\ell$  and  $s \in \tilde{L}_{\ell+1}$  if  $s$  is free in  $M$  and  $s$  is adjacent to  $u$ .
- We add an edge between  $(u, v) \in \tilde{L}_i$  and  $(u', v') \in \tilde{L}_{i+1}$  for  $i \in [\ell - 1]$  if  $v$  is adjacent to  $u'$ .

Note that inactive vertices are isolated in  $H$ .



■ **Figure 1** Example of an active layered graph with respect to a matching  $M$ , in solid lines. The free vertex  $5$  appears in  $L_0$  and the free vertex  $6$  appears in  $L_3$ . The augmenting path found by the layered graph is represented by a dashed purple line.

We introduce the following definition to handle augmenting paths for a matching  $M$  in a graph  $G$  via paths in its corresponding layered graph.

► **Definition 5.** We say that a path  $v_i, v_{i-1}, \dots, v_0$  with  $v_j \in \tilde{L}_j$  ( $j \in \{0, 1, \dots, i\}$ ) is an  $i$ -path. Note that an  $(\ell + 1)$ -path in  $H$  corresponds to an augmenting path of length  $2\ell + 1$  in  $G$ .

The layered graph defined above is slightly different from the original one due to McGregor [9] in that he did not include inactive vertices in  $H$ , as they are irrelevant to find augmenting paths. However as we consider sensitivity of algorithms, it is convenient to fix the vertex set so that it is independent of the current matching  $M$ .

We briefly define the randomized greedy subroutine **RANDOMIZEDGREEDY** on a graph  $G = (V, E)$  as follows. The subroutine first chooses a random ordering  $\pi$  over edges and then starting with an empty matching  $M$ , the procedure iteratively adds the  $i$ -th edge in the ordering  $\pi$  to  $M$  if the edge is not adjacent to any edge in  $M$ , until it has processed all edges. See Algorithm 1 for the full details.

■ **Algorithm 1** Randomized Greedy Algorithm.

---

```

1 Procedure RANDOMIZEDGREEDY(Graph  $G = (V, E)$ )
2   Generate a permutation  $\pi$  of  $E$  uniformly at random;
3   Greedily add edges to a maximal matching  $M$ , in the order of  $\pi$ ;
4   Output  $M$ ;

```

---

Algorithm 2 shows our algorithm for finding a large set of augmenting paths of length  $2\ell + 1$  given a matching  $M$  in a graph  $G$ . For a matching  $M$  and a vertex  $v$  belonging to an edge  $e$  in  $M$ , let  $\Gamma_M(v)$  denote the other endpoint of  $e$ . Similarly, for a vertex set  $S$  such that each edge in  $M$  uses at most one vertex in  $S$ , let  $\Gamma_M(S)$  denote the set of other endpoints. For subsets  $L$  and  $R$  of adjacent layers in  $H$ , let **RANDOMIZEDGREEDY**( $L, R$ ) denote the randomized greedy on the induced bipartite graph  $H[L \cup R]$ . **FINDPATHS** tries to find a large set of vertex disjoint  $i$ -paths from  $S \subseteq L_i$  to  $L_0$ . The result is stored as a tag function  $t : V(H) \rightarrow V(H) \cup \{\text{untagged}, \text{dead end}\}$ . Here,  $t(v)$  is initialized to **untagged**, and it will represent the next vertex in the  $i$ -path found. If we could not find any  $i$ -path starting from  $v$ ,  $t(v)$  is set to **deadend**.

The difference from McGregor's algorithm is that we run the loop in **FINDPATHS**  $1/\delta$  times instead of running it until  $|M'| \leq \delta|M|$ . This makes sure that we compute a maximal matching the same number of times no matter what  $G$  and  $M$  are, and it is more convenient when analyzing the sensitivity.

Our algorithm for the maximum matching problem (Algorithm 3) simply runs **AUGMENTINGPATHS** sufficiently many times for various choice of  $\ell$  and then keep applying the obtained augmenting paths. Before analyzing its approximation ratio and sensitivity, we analyze its running time.

► **Lemma 6.** *The total running time of Algorithm 3 is  $O(n + m)K$ , where  $K = (1/\epsilon)^{2^{O(1/\epsilon)}}$ .*

**Proof.** Observe that the outer loop of Algorithm 3 runs for  $r$  iterations and the inner loop runs for  $k$  iterations, where  $k = \lceil \epsilon^{-1} + 1 \rceil$  and  $r = 4k^2(16k + 20)(k - 1)(2k)^k$ . Each inner loop runs an instance of **AUGMENTINGPATHS** with parameters  $\ell \leq k$  and  $\delta = \frac{1}{r(2k+2)}$ , which creates a layered graph with  $O(\ell)$  layers in  $O((n + m)k)$  time, and then calls **FINDPATHS**. For each time that **FINDPATHS** is called, the value of  $\delta$  is squared and the value of  $\ell$  is decremented, starting at  $\ell = k$  until  $\ell = 1$ . Thus, the loop in **FINDPATHS** is run at most  $\frac{1}{\delta} = O\left((r(2k + 2))^{2^k}\right)$  times and each loop uses time  $O(n + m)$ . Hence, the total runtime is  $O(n + m)K$ , where  $K = (1/\epsilon)^{2^{O(1/\epsilon)}}$ . ◀

## 2.2 Approximation Ratio

In this section, we analyze the approximation ratio of Algorithm 3.

► **Lemma 7** ([9]). *Suppose **FINDPATHS**( $\cdot, \cdot, j, \cdot$ ) is called  $\delta^{-2^{i-j+1}+1}$  times in the recursion for **FINDPATHS**( $H, S, i, \delta$ ). Then at most  $2\delta|L_i|$  paths are removed from consideration as being  $(i + 1)$ -paths.*



---

**Algorithm 2** Augmentation Algorithm.
 

---

```

1 Procedure AUGMENTINGPATHS( $G, M, \ell, \delta$ )
2   Construct a layered graph  $H$  using  $G, M$ , and  $\ell$ ;
3    $t(v) \leftarrow \text{untagged}$  for every vertex  $v$  in  $H$ ;
4   FINDPATHS( $H, L_{\ell+1}, \ell, \delta, t$ );
5   Convert  $t$  to a set of augmenting paths in  $G$ ;
6   Apply the augmenting paths to  $M$ .

7 Procedure FINDPATHS( $H, S, i, \delta, t$ )
8    $M' \leftarrow \text{RANDOMIZEDGREEDY}(S, L_{i-1} \cap t^{-1}(\text{untagged}))$ ;
9    $S' \leftarrow \Gamma_{M'}(S)$ ;
10  if  $i = 1$  then
11    for  $u \in S$  do
12      if  $u \in \Gamma_{M'}(L_0)$  then
13         $t(u) \leftarrow \Gamma_{M'}(u)$ 
14      if  $u \in S \setminus \Gamma_{M'}(L_0)$  then
15         $t(u) \leftarrow \text{dead end}$ .
16    return  $t$ .
17  for  $\lceil 1/\delta \rceil$  times do
18    FINDPATHS( $H, S', i-1, \delta^2, t$ );
19    for  $v \in S' \setminus t^{-1}(\text{dead end})$  do
20       $t(\Gamma_{M'}(v)) \leftarrow v$ .
21     $M' \leftarrow \text{RANDOMIZEDGREEDY}(S \cap t^{-1}(\text{untagged}), L_{i-1} \cap t^{-1}(\text{untagged}))$ ;
22     $S' \leftarrow \Gamma_{M'}(S \cap t^{-1}(\text{untagged}))$ .
23  for  $v \in S \cap t^{-1}(\text{untagged})$  do
24     $t(v) \leftarrow \text{dead end}$ .
25  return

```

---

**Algorithm 3** Algorithm for maximum matching.
 

---

```

1 Procedure MATCHING( $G, \epsilon$ )
2   Let  $\pi$  be a random ordering of the edges of  $G$ ;
3   Let  $M$  be the greedy maximal matching of  $G$  induced by  $\pi$ ;
4    $k \leftarrow \lceil \epsilon^{-1} + 1 \rceil$ ;
5    $r \leftarrow 4k^2(16k + 20)(k - 1)(2k)^k$ ;
6   for  $\ell = 1$  to  $k$  do
7     for  $i = 1$  to  $r$  do
8        $M'_{\ell,i} \leftarrow \text{AUGMENTINGPATHS}(G, M, \ell, \frac{1}{r(2k+2)});$ 
9        $M \leftarrow M \oplus M'_{\ell,i}$ .
10  return  $M$ .

```

---

Let  $\mathcal{L}_i$  be the set of graphs whose vertices are partitioned into  $i + 2$  layers,  $L_0, \dots, L_{i+1}$  and whose edges are a subset of  $\bigcup_{j=1}^{i+1} (L_j \times L_{j-1})$ . Then we immediately have the following lemma, analogous to Lemma 2 in [9]:

► **Lemma 8.** For a graph  $G \in \mathcal{L}_i$ ,  $\text{FINDPATHS}(H, S, i, \delta)$  finds at least  $(\gamma - \delta)|M|$  of the  $(i + 1)$ -paths among some maximal set of  $(i + 1)$ -paths of size  $\gamma|M|$ .

We require the following structural property relating maximal and maximum matchings through the set of connected components in the symmetric difference.

► **Lemma 9** (Lemma 1 in [9]). Let  $M$  be a maximal matching and  $M^*$  be a maximum matching. Let  $\mathcal{C}$  be the set of connected components in  $M^* \triangle M$ . Let  $\alpha_\ell$  be the constant so that  $\alpha_\ell|M|$  is the number of connected components in  $\mathcal{C}$  with  $\ell$  edges from  $M$ , excluding those with  $\ell$  edges from  $M^*$ . If  $\max_{\ell \in [k]} \alpha_\ell \leq \frac{1}{2k^2(k+1)}$ , then  $|M| \geq \frac{|M^*|}{1+1/k}$ .

We also require the following result by [9] bounding the number of augmenting paths found by  $\text{AUGMENTINGPATHS}$ .

► **Lemma 10** (Theorem 1 in [9]). If  $G$  has  $\alpha_\ell|M|$  augmenting paths of length  $2\ell + 1$ , then the number of augmenting paths of length  $2\ell + 1$  found by  $\text{AUGMENTINGPATHS}$  is at least  $(b_\ell\beta_\ell - \delta)|M|$ , where  $b_\ell = \frac{1}{2\ell+1}$  and  $\beta_\ell \sim \mathcal{B}\left(\alpha_\ell|M|, \frac{1}{2(2\ell)^\ell}\right)$ .

We now show that Algorithm 3 outputs a  $(1 - \epsilon)$ -approximation to the maximum matching.

► **Theorem 11.** Algorithm 3 finds a  $(1 - \epsilon)$ -approximation to the maximum matching with probability at least 0.99.

**Proof.** We say the algorithm enters *phase*  $\ell$  when the number of layers in the layered graph has been incremented to  $\ell$ , i.e., each invocation of the outer for loop corresponds to a separate phase. We say the algorithm enters *round*  $i$  in phase  $\ell$  after the subroutine  $\text{AUGMENTINGPATHS}$  has completed  $i - 1$  iterations within phase  $\ell$ . Let  $M_{\ell,i}$  be the matching  $M$  prior to the call to  $\text{AUGMENTINGPATHS}$  in round  $i$  of phase  $\ell$ . Let  $\alpha_{\ell,i}|M_{\ell,i}|$  be the number of length  $2\ell + 1$  augmenting paths of  $M_{\ell,i}$ . Thus by Lemma 10, the subroutine  $\text{AUGMENTINGPATHS}$  augments  $M_{\ell,i}$  by at least  $(b_\ell\beta_{\ell,i} - \delta)|M_{\ell,i}|$  edges in round  $i$  of phase  $\ell$ , where  $b_\ell = \frac{1}{2\ell+2}$ ,  $\delta$  is a parameter that we choose later, and  $\beta_{\ell,i}$  is a random variable distributed according to  $\mathcal{B}\left(\alpha_{\ell,i}|M_{\ell,i}|, \frac{1}{2(2\ell)^\ell}\right)$ . Let  $M$  be the output matching. Then by Bernoulli's inequality, we have

$$\begin{aligned} \Pr[|M| \geq 2|M_{1,1}|] &\geq \Pr\left[|M_{1,1}| \prod_{\ell \in [k], i \in [r]} (1 + \max(0, b_\ell\beta_{\ell,i} - \delta)) \geq 2|M_{1,1}|\right] \\ &\geq \Pr\left[\sum_{\ell \in [k]} \max_{i \in [r]} b_\ell\beta_{\ell,i} \geq 2 + r\delta\right]. \end{aligned}$$

We would like to analyze  $\sum_{\ell \in [k]} \max_{i \in [r]} b_\ell\beta_{\ell,i}$ , but the analysis is challenging due to dependencies between multiple rounds and phases. We thus define independent variables  $X_1, \dots, X_k$  and use a coupling argument.

We define  $A_\ell = \max_{i \in [r]} b_\ell\beta_{\ell,i}|M_{\ell,i}|$  to be an upper bound on the maximum number of augmented edges during phase  $\ell$  of the algorithm. Suppose by way of contradiction that  $\max_{i \in [r]} \alpha_{\ell,i} < \alpha_0 := \frac{1}{2k^2(k-1)}$  for each of the phases  $1 \leq \ell \leq k$ . Then Lemma 9 would imply that at some point  $M_{\ell,i}$  is sufficiently large. Thus we have  $\max_{i \in [r]} \alpha_{\ell,i} \geq \alpha_0$ .

We have  $A_\ell = \max_{i \in [r]} b_\ell\beta_{\ell,i}|M_{\ell,i}|$ ,  $b_\ell = \frac{1}{2\ell+2}$ , and  $\beta_{\ell,i}|M_{\ell,i}| \sim \mathcal{B}\left(\alpha_{\ell,i}|M_{\ell,i}|, \frac{1}{2(2\ell)^\ell}\right)$ . Now for  $\ell \leq k$ , we have that  $\frac{1}{2(2\ell)^\ell} \geq \frac{1}{2(2k)^k}$ . Thus  $\max_{i \in [r]} \alpha_{\ell,i} \geq \alpha_0$  and  $|M_{\ell,i}| \geq |M_{\ell,1}|$  implies that the distribution of  $A_\ell$  statistically dominates the distribution of  $b_\ell \cdot \mathcal{B}\left(\alpha_0|M_{\ell,1}|, \frac{1}{2(2k)^k}\right)$ .

Hence if we define  $X_\ell$  to be independent random variables distributed as  $\mathcal{B}\left(\alpha_0|M_{\ell,1}|, \frac{1}{2(2k)^k}\right)$  for each  $\ell \in [r]$ , then the distribution of  $A_\ell$  statistically dominates the distribution of  $b_\ell \cdot X_\ell$ . Thus,

$$\begin{aligned} \Pr \left[ \sum_{\ell \in [k]} \max_{i \in [r]} b_\ell \beta_{\ell,i} \geq 2 + r\delta \right] &\geq \Pr \left[ \sum_{\ell \in [k]} \max_{i \in [r]} b_\ell \beta_{\ell,i} |M_{\ell,i}| \geq (2 + r\delta)|M_{k,r}| \right] \\ &= \Pr \left[ \sum_{\ell \in [k]} A_\ell \geq (2 + r\delta)|M_{k,r}| \right] \\ &\geq \Pr \left[ \sum_{\ell \in [k]} X_\ell \geq \frac{4 + 2r\delta}{b_k} \cdot |M_{1,1}| \right], \end{aligned}$$

where the final inequality results from  $M_{1,1}$  being a maximal matching and  $b_\ell \geq b_k$  for  $\ell \in [k]$ .

Now the variables  $X_\ell$  are independent but not identically distributed. Nevertheless, we can write  $Y = \sum_{\ell \in [k]} X_\ell$  and note that the distribution of  $Y$  statistically dominates the distribution of  $Z \sim \mathcal{B}\left(\alpha_0|M_{1,1}|r, \frac{1}{2(2k)^k}\right)$  since  $|M_{\ell,i}| \geq |M_{1,1}|$  for all  $\ell \in [k]$  and  $i \in [r]$ . Thus for  $b_k = \frac{1}{2k+2}$  and  $\delta = \frac{b_k}{r}$ ,

$$\begin{aligned} \Pr \left[ \sum_{\ell \in [k]} X_\ell \geq \frac{4 + 2r\delta}{b_k} \cdot |M_{1,1}| \right] &\geq \Pr \left[ Z \geq \frac{4 + 2b_k}{b_k} |M_{1,1}| \right] \\ &= \Pr[Z \geq |M_{1,1}|(8k + 10)]. \end{aligned}$$

For  $r = \frac{2(2k)^k(16k+20)}{\alpha_0}$ , we have that  $\mathbf{E}[Z] = (16k + 20)|M_{1,1}|$ . Thus from a simple Chernoff bound,

$$\Pr[Z \geq |M_{1,1}|(8k + 10)] = 1 - \Pr \left[ Z < \frac{\mathbf{E}[Z]}{2} \right] > 1 - e^{-2(16k+20)|M_{1,1}|} \geq 0.99.$$

Putting things together, we have  $\Pr[|M| \geq 2|M_{1,1}|] \geq 0.99$ , which implies that there exists a maximum matching with more than double the number of edges of a maximal matching and is a contradiction. Therefore, our assumption that  $\max_{i \in [r]} \alpha_{\ell,i} \geq \alpha_0 := \frac{1}{2k^2(k-1)}$  for each of the phases  $1 \leq \ell \leq k$  must have been invalid. However, if  $\alpha_{\ell,i} < \frac{1}{2k^2(k-1)}$  for some  $i \in [r]$  and  $\ell \in [k]$ , then by Lemma 9, we have for  $k = \lceil \epsilon^{-1} + 1 \rceil$  and sufficiently small  $\epsilon > 0$  that

$$|M_{\ell,i}| \geq \frac{|M^*|}{1 + 1/k} \geq \frac{|M^*|}{1 + \epsilon} \geq (1 - \epsilon)|M^*|,$$

with probability at least 0.99. Thus, Algorithm 3 outputs a  $(1 - \epsilon)$  approximation the maximum matching with probability at least 0.99.  $\blacktriangleleft$

**Boosting the Success Probability.** To increase the probability of success to  $1 - p$  for any  $p \in (0, 1)$ , a naïve approach would be to run  $O\left(\log \frac{1}{p}\right)$  iterations of Algorithm 3 in parallel. However, the sensitivity analysis becomes considerably more challenging. Instead, we note that the  $\frac{1}{e}$  probability of failure is actually a significant weakening of the  $e^{-2(16k+20)|M_{1,1}|}$  probability of failure. Thus, increasing  $k$  by a factor of  $O\left(\log \frac{1}{p}\right)$  increases the probability of success to  $1 - p$ . However, for subconstant  $p$ , it also substantially increases the asymptotic sensitivity of Algorithm 3.

### 2.3 Sensitivity of the Randomized Greedy and Algorithm 3

To analyze the sensitivity of Algorithm 3, we first analyze the sensitivity of the randomized greedy algorithm.

#### 2.3.1 Sensitivity of the Randomized Greedy

In this section, we study the sensitivity of the randomized greedy with respect to vertex deletions. Recall that given a graph  $G = (V, E)$ , the randomized greedy works as follows. First, it chooses a random ordering  $\pi$  over edges. Then starting with an empty matching  $M$ , it iteratively adds the  $i$ -th edge in the ordering  $\pi$  to  $M$  if it is not adjacent to any edge in  $M$ . The main result of this section is the following.

► **Theorem 12.** *Let  $A$  be the randomized greedy for the maximum matching problem. Then, for any graph  $G = (V, E)$  and a vertex  $v \in V$ , we have  $d_{\text{EM}}(A(G), A(G - v)) \leq 1$ .*

We need to consider deleting vertices to analyze the sensitivity of our randomized  $(1 - \epsilon)$ -approximation algorithm for the maximum matching problem in Section 2. Our analysis is a slight modification of a similar result for the maximal independent set problem [2]. Hence, we defer the proof to the full version of the paper.

#### 2.3.2 Sensitivity of Algorithm 3

We first analyze the sensitivity of AUGMENTINGPATHS. Let us fix the graphs  $G = (V, E)$  and  $G' = (V, E')$ , and matchings  $M \subseteq E$  and  $M' \subseteq E'$ , a positive integer  $\ell$ , and  $\delta > 0$ . Let  $H$  and  $H'$  be the layered graphs constructed using  $G, M, \ell$  and  $G', M', \ell$ , respectively, and let  $\tilde{L}$  and  $\tilde{L}'$  be the set of active vertices in  $H$  and  $H'$ , respectively.

► **Lemma 13.** *We have  $d_H(\tilde{L}, \tilde{L}') \leq 3d_H(E, E') + 3d_H(M, M')$ .*

**Proof.** Each edge modification in the graph or the matching may cause activate/inactivate at most three vertices in the layered graph (two of them are in the first and last layers, and the remaining one is in one of the middle layers), and hence the lemma follows. ◀

For two tag functions  $t, t': V(H) \rightarrow V(H) \cup \{\text{untagged}, \text{deadend}\}$ , we define  $d_H(t, t') = |\{v \in V(H) \mid t(v) \neq t'(v)\}|$ . We will use symbols  $t$  and  $t'$  to denote tag functions for  $H$  and  $H'$ , respectively. Note that the supposed domain of  $t'$  is  $V(H')$ , but it is equal to  $V(H)$ .

► **Lemma 14.** *Let  $\mathcal{P} = \text{AUGMENTINGPATHS}(G, M, \ell, \delta)$  and  $\mathcal{P}' = \text{AUGMENTINGPATHS}(G', M', \ell, \delta)$ . Then, we have*

$$d_{\text{EM}}(\mathcal{P}, \mathcal{P}') \leq (d_H(E, E') + d_H(M, M')) \cdot 3^K,$$

where  $K = (1/\epsilon)^{2^{O(1/\epsilon)}}$ .

**Proof.** Let  $A$  and  $A'$  denote  $\text{AUGMENTINGPATHS}(G, M, \delta, \ell)$  and  $\text{AUGMENTINGPATHS}(G', M', \delta, \ell)$ , respectively. Let  $M_1, M_2, \dots, M_K$  and  $M'_1, M'_2, \dots, M'_K$  be the sequences of matchings constructed during the process of  $A$  and  $A'$ , respectively. Note that  $A$  and  $A'$  construct the same number of matchings, and that  $K \leq (1/\epsilon)^{2^{O(1/\epsilon)}}$ , which follows by a similar argument to that in the proof of Lemma 6. For  $i \in [K]$ , let  $S_i$  and  $S'_i$  be the vertex sets on which  $M_i$  and  $M'_i$ , respectively, are constructed, that is, the vertex set passed on to RANDOMIZEDGREEDY, and let  $t_i$  and  $t'_i$  be the tag functions right before constructing  $M_i$  and  $M'_i$ , respectively. By Lemma 13, we have  $d_{\text{EM}}(S_1, S'_1) \leq c$ , where  $c = 3d_H(M, M') + 3d_H(E, E')$ . First, because each difference between  $M_{i-1}$  and  $M'_{i-1}$  increases the Hamming distance between  $t_i$  and  $t'_i$  by one, we have

$$\begin{aligned}
d_{\text{EM}}(t_i, t'_i) &\leq d_{\text{EM}}(M_{i-1}, M'_{i-1}) + d_{\text{EM}}(t_{i-1}, t'_{i-1}) \\
&\leq \sum_{j=1}^{i-1} d_{\text{EM}}(M_j, M'_j) + d_{\text{EM}}(t_1, t'_1) = \sum_{j=1}^{i-1} d_{\text{EM}}(M_j, M'_j).
\end{aligned}$$

Then we have

$$d_{\text{EM}}(S_i, S'_i) \leq d_{\text{EM}}(M_{i-1}, M'_{i-1}) + d_{\text{EM}}(t_i, t'_i) \leq 2 \sum_{j=1}^{i-1} d_{\text{EM}}(M_j, M'_j) \leq 2 \sum_{j=1}^{i-1} d_{\text{EM}}(S_j, S'_j),$$

where the last inequality is due to Theorem 12. Solving this recursion, we get

$$\sum_{j=1}^i d_{\text{EM}}(S_j, S'_j) \leq c \cdot 3^{i-1},$$

and hence we have  $d_{\text{EM}}(S_K, S'_K) = 2 \cdot 3^{K-2}c$ , and the claim follows.  $\blacktriangleleft$

We now show that the sensitivity of Algorithm 3 is  $O_\epsilon(1)$ .

► **Theorem 15.** *The sensitivity of Algorithm 3 is at most  $3^K$ , where  $K = (1/\epsilon)^{2^{O(1/\epsilon)}}$ .*

**Proof.** Let  $G = (V, E)$  be a graph and  $G' = (V, E') = G - e$  for some  $e \in E$ . Let  $M_0, M_1, \dots, M_{kr}$  be the sequence of matchings we construct in Algorithm 3 on  $G$ , where  $M_0$  is the matching constructed at Line 3, and  $M_j$  is the matching constructed at Line 3 in the round  $i$  of the phase  $\ell$  such that  $j = (\ell - 1)r + i$ . We define  $M'_0, M'_1, \dots, M'_{kr}$  similarly using  $G'$ . Then, we have by Theorem 12

$$d_{\text{EM}}(M_0, M'_0) \leq 1,$$

and we have by Lemma 14

$$\begin{aligned}
d_{\text{EM}}(M_i, M'_i) &\leq d_{\text{EM}}(M_{i-1}, M'_{i-1}) + (d_{\text{EM}}(E, E') + d_{\text{EM}}(M_{i-1}, M'_{i-1})) \cdot 3^K \\
&= d_{\text{EM}}(M_{i-1}, M'_{i-1})(3^K + 1) + 3^K
\end{aligned}$$

for  $i \in [kr]$ , where  $K = (1/\epsilon)^{2^{O(1/\epsilon)}}$ . Solving the recursion, we get

$$d_{\text{EM}}(M_{kr}, M'_{kr}) \leq 2(1 + 3^K)^{kr} - 1,$$

and we have the desired bound.  $\blacktriangleleft$

The proof of Theorem 1 then follows from Theorem 11 and Theorem 15.

### Sensitivity to Vertex Deletions

We remark that Algorithm 3 also has sensitivity  $O(3^K)$ , for  $K = (1/\epsilon)^{2^{O(1/\epsilon)}}$ , to vertex deletions. Recall that Lemma 13 crucially relies on each edge deletion changing at most three vertices in the layered graph. That is, due to the construction of the layered graph, each edge deletion changes at most two altered vertices in the first and last layers, and at most one altered vertex in one of the middle layers. This is because the first layer and the last layer encode the vertex set  $V$ , while each matched edge is assigned to one of the middle layers.

Observe that when we delete a vertex  $v$ , at most one vertex in the vertex set  $V$  is altered, so that the first and last layer of the layered graph each have one change. Moreover, at most one matched edge is incident to  $v$ , so at most one vertex in one of the middle layers is altered as well. Thus, at most three vertices in the layered graph are changed as a result of the vertex deletion, so the sensitivity of Algorithm 3 to vertex deletions is again  $O(3^K)$ , for  $K = (1/\epsilon)^{2^{O(1/\epsilon)}}$ .

### 3 Deterministic Maximal Matching for Bounded-Degree Graphs

In this section, we give a deterministic algorithm for computing a maximal matching that has low sensitivity on bounded-degree graphs. The main idea is to use deterministic local computation algorithms with a small number of probes to find a maximal matching. Our algorithm uses two main ingredients. The first ingredient is a deterministic LCA of [4] for  $6^\Delta$ -coloring a graph with maximum degree  $\Delta$ , using  $O(\Delta \log^* n)$  probes. The second ingredient is a framework of [10] that simulates local distributed algorithms using a deterministic LCA. In particular, we use the framework to simulate an algorithm that takes a coloring of a graph and outputs a maximal matching. We give the details for the local distributed algorithm in Algorithm 5.

To bound the sensitivity of the algorithm, it suffices to analyze the number of queries for which a deleted edge would be probed. Crucially, both the deterministic LCA of [4] and the framework of [10] only probe edges (incident to vertices) within a small radius of the query. Thus, only a small number of queries will probe the edge that is altered, so that the output of the algorithm only has a small number of changes.

We first require a deterministic LCA of [4] for  $6^\Delta$ -coloring a graph with degree  $\Delta$ , using a small number of probes within distance  $O(\Delta \log^* n)$  of the query. We give the full details in Algorithm 4, which has the following guarantee.

► **Lemma 16** ([4]). *There exists a deterministic LCA  $\text{COLORINGLCA}$  for  $6^\Delta$ -coloring a graph with degree  $\Delta$ , using  $O(\Delta \log^* n)$  probes.*

We now describe a local distributed algorithm that takes a coloring of a graph and outputs a maximal matching. The algorithm iterates over all colors and adds any edge adjacent to a vertex of a particular color to the greedy matching if there is no other adjacent edge already present in the matching. We give the algorithm in full in Algorithm 5.

Putting things together, we obtain a deterministic maximal matching algorithm in Algorithm 6.

We next require the following framework of [10] that simulates local distributed algorithms using a deterministic LCA. In particular, we will implement Algorithm 5.

► **Lemma 17** ([7, 10]). *Given access to an oracle that takes vertices of an underlying graph as queries and outputs a color for the queried vertex, there exists a deterministic LCA that can implement  $\text{COLORING-TO-MM}$  using  $\Delta^{O(c)}$  probes.*

Given a  $\text{COLORINGLCA}$  for  $6^\Delta$ -coloring, the LCA for maximal matching in Lemma 17 uses the following idea. For a query edge  $e$ , we first call  $\text{COLORINGLCA}$  for every vertex with distance roughly  $6^\Delta$  from  $e$ . Parnas and Ron [10] then shows it suffices to run Algorithm 5 locally on the graph of radius roughly  $6^\Delta$  from  $e$ .

We now show that our deterministic LCA based algorithm outputs a maximal matching with low worst case sensitivity for low-degree graphs.

■ **Algorithm 4** LCA Algorithm COLORINGLCA for  $6^\Delta$ -coloring with  $O(\Delta \log^* n)$  probes.

---

```

1 Procedure FORMFORESTS( $G, \Delta$ )
2   //Decompose graph into  $\Delta$  oriented forests;
3   for  $i = 1$  to  $\Delta$  do
4     Let  $N_u(i)$  denote the  $i$ -th neighbor of  $u$  according to the IDs of the vertices;
5     Let  $E_i = \{(u, v) : \text{id}(u) < \text{id}(v), v = N_u(i)\}$ ;
6     Let  $G_i = (V, E_i)$  be the oriented tree, where the root has no out-going edges;
7 Procedure COLORFORESTS( $G_i, \Delta$ )
8   for  $\Theta(\log^* n)$  rounds do
9     for each nodes  $u$  do
10      if  $u$  is a root node then
11        Set  $\phi_u$  to 0;
12      else
13        Let  $v$  be the parent of  $u$  in  $G_i$ ;
14        Let  $a_u$  be the index of the least significant bit with  $\phi_u \neq \phi_v$ ;
15        Let  $b_u$  be the value of the  $a_u$ -th bit of  $v$ ;
16         $\phi_u \leftarrow a_u \circ b_u$ .
```

---

■ **Algorithm 5** Maximal Matching Algorithm COLORING-TO-MM.

---

```

1 Procedure COLORING-TO-MM( $G$  colored with  $c$  colors)
2    $M \leftarrow \emptyset$ ;
3   for color  $i = 1$  to  $i = c$  do
4     if edge  $(u, v)$  has either  $\phi_u = i$  or  $\phi_v = i$  then
5       Add  $(u, v)$  to  $M$  if no adjacent edge is in  $M$ .
```

---

■ **Algorithm 6** Maximal Matching Algorithm.

---

```

1 Procedure COLORING-TO-MM(Graph  $G$ )
2   Coloring  $G_C = \text{COLORINGLCA}(G, \Delta)$ ;
3   Output COLORING-TO-MM( $G_C$ );
```

---

**Proof of Theorem 2.** Consider running the deterministic LCA from Lemma 17 that simulates COLORING-TO-MM on a graph  $G$  and a graph  $G' := G - e$ , for some  $e \in E$ . Let  $S$  be the set of vertices that are assigned different colors in  $G$  and  $G'$  by COLORINGLCA. First observe that  $e$  is within distance  $\Theta(\log^* n)$  from at most  $\Delta^{\Theta(\log^* n)}$  other vertices. Hence, from Lemma 16 we have  $|S| \leq \Delta^{\Theta(\log^* n)}$ . Moreover, each vertex  $u \in S$  is within distance  $O(6^\Delta)$  from at most  $\Delta^{O(6^\Delta)}$  other vertices. Thus the total number of edges that differ between the matchings  $M$  and  $M'$  output by COLORING-TO-MM for  $G$  and  $G'$  respectively is at most

$$\Delta^{\Theta(\log^* n)} \cdot \Delta^{O(6^\Delta)} = \Delta^{O(6^\Delta + \log^* n)}.$$

◀

It is clear that an almost identical analysis goes through for vertex sensitivity.



## 4 Lower Bounds for Maximum Matching

In this section, we show lower bounds for deterministic and randomized algorithms for the maximum matching problem.

### 4.1 Deterministic Lower Bound

In this section, we prove Theorem 3, which claims that *any* deterministic algorithm for the maximum matching problem has edge sensitivity  $\Omega(\log^* n)$ . Our proof relies on Ramsey's theorem. First we introduce some definitions. Let  $Y$  be a finite set. We say that  $X$  is a  $k$ -subset of  $Y$  if  $X \subseteq Y$  and  $|X| = k$ . Let  $Y^{(k)} = \{X \subseteq Y \mid |X| = k\}$  be the collection of all  $k$ -subsets of  $Y$ . A  $c$ -labeling of  $Y^{(k)}$  is an arbitrary function  $f : Y^{(k)} \rightarrow [c]$ . Then we say that  $X \subseteq Y$  is *monochromatic* in  $f$  if  $f(A) = f(B)$  for all  $A, B \in X^{(k)}$ . Let  $R_c(n; k)$  be the smallest integer  $N$  such that the following holds: for any set  $Y$  with at least  $N$  elements, and for any  $c$ -labeling  $f$  of  $Y^{(k)}$ , there is an  $n$ -subset of  $Y$  that is monochromatic in  $f$ . If no such  $N$  exists,  $R_c(n; k) = \infty$ . Define  $\text{twr}(k)$  as the tower of twos of height  $k$ , that is,  $\text{twr}(1) = 2$  and  $\text{twr}(k+1) = 2^{\text{twr}(k)}$ . We will use the following formulation of Ramsey's theorem.

► **Theorem 18** (Special case of Theorem 1 in [6]). *For any positive integer  $t$ ,  $R_{2^t}(t+1; t) \leq \text{twr}(O(t))$ .*

We now show that any deterministic constant-factor approximation algorithm for the maximum matching problem has edge sensitivity  $\Omega(\log^* n)$ .

**Proof of Theorem 3.** Let  $A$  be an arbitrary deterministic algorithm that outputs a maximal matching, let  $t$  be a positive integer, which will be determined later. Let  $\mathcal{G}$  be a class of graphs on the vertex set  $[n]$  consisting of a cycle  $v_1, \dots, v_t$  with  $v_1 < v_2 < \dots < v_t$  and  $n-t$  isolated vertices. Given a matching  $M$  on the cycle  $v_1, \dots, v_t$ , we encode it to an integer  $0 \leq k \leq 2^t - 1$  so that the  $i$ -th bit of  $k$  is 1 if and only if the edge  $\{v_i, v_{i+1}\}$  belongs to  $M$ , where we regard  $v_{t+1} = v_1$ . Then, we can regard the algorithm  $A$  as a function  $f : \binom{[n]}{t} \rightarrow \{0, 1, \dots, 2^t - 1\}$ , that is, given a set  $\{v_1, \dots, v_t\} \subseteq [n]$  with  $v_1 < v_2 < \dots < v_t$ , we compute a matching on the cycle  $v_1, \dots, v_t$ , and encode it to an integer. Then if  $n \geq R_{2^t}(t+1; t)$ , which holds when  $t = O(\log^* n)$  by Theorem 18, there exists a set  $S = \{s_0, s_1, \dots, s_t\} \subseteq [n]$  with  $s_0 < s_1 < \dots < s_t$  such that  $f(T)$  is constant whenever  $T \subseteq S$  with  $|T| = t$ . Let  $G, G' \in \mathcal{G}$  be the graph with cycles  $s_0, \dots, s_{t-1}$  and  $s_1, \dots, s_t$ , respectively, and let  $M$  and  $M'$  be the matching output by  $A$  on  $G$  and  $G'$ , respectively. As  $M$  and  $M'$  have the same encoding,  $\{s_i, s_{i+1 \bmod t}\} \in M$  if and only if  $\{s_{i+1}, s_{i+2}\} \in M'$ , where we regard  $s_{t+2} = s_1$ . Note that, however, if  $\{s_i, s_{i+1 \bmod t}\} \in M$  then  $\{s_{i+1 \bmod t}, s_{i+2 \bmod t}\} \notin M'$ . It follows that  $d_H(M, M') = \Omega(|M|) = \Omega(t)$ , where the last equality holds because  $A$  has a constant approximation ratio. ◀

### 4.2 Lower Bounds for Deterministic Greedy Algorithm

As we have seen in Section 2.3.1, the randomized greedy algorithm has  $O(1)$  sensitivity even for vertex deletion. Can we derandomize it without increasing the sensitivity? To make the question more precise, let  $V$  be a set of  $n$  vertices and  $\pi$  be a permutation over  $\binom{V}{2}$ . Then, let  $A_\pi$  denote the greedy algorithm such that, starting with an empty matching  $M$ , it iteratively adds the  $i$ -th edge with respect to  $\pi$  to  $M$  if and only if the edge does not share an endpoint with any edge in  $M$ . We now show that the answer to the question is negative.

► **Theorem 19.** *For any permutation  $\pi$  over  $\binom{V}{2}$ , the algorithm  $A_\pi$  has sensitivity  $\Omega(n)$ .*

**Proof.** We say that an element  $e$  of a poset *covers* another element  $e'$  if  $e > e'$ , where  $>$  is the order relation of the poset, and there is no other element  $e''$  such that  $e > e'' > e'$ . Then, we construct a poset  $P$  on the element set  $\binom{V}{2}$  in which a pair  $e \in \binom{V}{2}$  covers another pair  $e' \in \binom{V}{2}$  if  $\pi(e) > \pi(e')$  and  $|e \cap e'| \geq 1$ . Note that the size of any antichain in  $P$  is at most  $n/2$ : A set of elements of size more than  $n/2$  must have two elements  $e, e'$  with  $|e \cap e'| \geq 1$ , which form a chain of length two. Hence, we need at least  $\binom{n}{2}/(n/2) = n - 1$  antichains to cover all the elements in  $P$ . Then by Mirsky's theorem, there exists a chain, say,  $e_1, \dots, e_{n-1}$ , of size  $n - 1$  in  $P$ . From the construction of  $P$ ,  $e_1, \dots, e_{n-1}$  forms a path of length  $n - 1$ . Then,  $A_\pi$  on the path  $e_1, \dots, e_{n-1}$  outputs edges with odd indices, whereas  $A_\pi$  on the path  $e_2, \dots, e_{n-1}$  outputs edges with even indices, and hence the sensitivity of  $A_\pi$  is  $\Omega(n)$ . ◀

### 4.3 Lower Bounds for Randomized Algorithms

The following shows that sensitivity must increase as approximation ratio goes to one.

► **Theorem 20.** *Let  $\epsilon > 0$ . Any (possibly randomized)  $(1 - \epsilon)$ -approximation algorithm for the maximum matching problem has sensitivity  $\Omega(1/\epsilon)$ .*

**Proof.** For simplicity, we assume  $1/10\epsilon$  is an even integer. Let  $A$  be an arbitrary  $(1 - \epsilon)$ -approximation algorithm for the maximum matching problem, and let  $G$  be a graph consisting of a cycle of length  $1/10\epsilon$  and  $n - 1/10\epsilon$  isolated vertices. Clearly  $G$  has two disjoint maximum matchings, say,  $M_1, M_2$ , of size  $1/20\epsilon$ . Let  $p_1$  and  $p_2$  be the probability that  $A$  on  $G$  outputs  $M_1$  and  $M_2$ , respectively. Then as  $A$  has approximation ratio  $1 - \epsilon$ , we have

$$p_1 \cdot \frac{1}{20\epsilon} + p_2 \cdot \frac{1}{20\epsilon} + (1 - p_1 - p_2) \cdot \left( \frac{1}{20\epsilon} - 1 \right) \geq \frac{1 - \epsilon}{20\epsilon}.$$

Hence, we have  $p_1 + p_2 \geq 19/20$ , and it follows that at least one of  $p_1 \geq 19/40$  and  $p_2 \geq 19/40$  hold. Without loss of generality, we assume  $p_1 \geq 19/40$ .

Let  $G'$  be the graph obtained from  $G$  by removing one edge in  $M_1$ . Then,  $G'$  has a unique maximum matching  $M_2$ . Let  $p'_2$  be the probability that  $A$  on  $G'$  outputs  $M_2$ . As  $A$  has approximation ratio  $1 - \epsilon$ , we have

$$p_2 \cdot \frac{1}{20\epsilon} + (1 - p_2) \cdot \left( \frac{1}{20\epsilon} - 1 \right) \geq \frac{1 - \epsilon}{20\epsilon},$$

which implies  $p'_2 \geq 19/20$ . Hence, the sensitivity of  $A$  is at least

$$\max\left(\Pr[A(G) = M_1] - \Pr[A(G') \neq M_2], 0\right) \cdot d_H(M_1, M_2) \geq \left(\frac{19}{40} - \frac{1}{20}\right) \cdot \frac{1}{10\epsilon} = \Omega\left(\frac{1}{\epsilon}\right). \blacktriangleleft$$

## 5 Weighted Sensitivity and Maximum Weighted Matching

In this section, we consider a generalization of sensitivity to weighted graphs, and show an approximation algorithm with low sensitivity for the maximum weighted matching problem.

### 5.1 Weighted Sensitivity

Given a weight function over the edges  $w : E \rightarrow \mathbb{R}$  of a graph  $G = (V, E)$  and two edge sets  $S$  and  $S'$ , let

$$d_H^w(S, S') = \sum_{e \in S \Delta S'} w(e),$$

where  $\triangle$  again denotes the symmetric difference. For random edge sets  $X$  and  $X'$ , we use  $d_{\text{EM}}^w(X, X')$  to denote the weighted earth mover's distance between  $X$  and  $X'$  with respect to  $w$ , so that

$$d_{\text{EM}}^w(X, X') = \min_{\mathcal{D}} \mathbf{E}_{(S, S') \sim \mathcal{D}} d_{\text{H}}^w(S, S'),$$

be the weighted Hamming distance between  $S$  and  $S'$  with respect to  $w$ , where  $\mathcal{D}$  is a distribution such that its marginal distributions on the first and second coordinates are  $X$  and  $X'$ , respectively. For a real-valued function  $\beta$  on graphs, we say that the *weighted sensitivity* of an algorithm  $A$  that outputs a set of edges is at most  $\beta$  if for every graph  $G = (V, E)$ , a weight function  $w : E \rightarrow \mathbb{R}$ , and an edge  $e \in E$ ,

$$d_{\text{EM}}^w(A(G), A(G - e)) \leq \beta(G).$$

A priori, it is not clear whether the weighted sensitivity of an algorithm should correlate with the weight of removed edges. Thus we say that the *normalized weighted sensitivity* is at most  $\beta$  if

$$\frac{d_{\text{EM}}^w(A(G), A(G - e))}{w(e)} \leq \beta(G).$$

## 5.2 Algorithm Description

We use a simple approach of partitioning the input by weight, finding a maximal matching on each partition, and finally forming a weighted matching by greedily adding edges from the maximal matchings, beginning with the matchings in the largest weight classes. The approach is known to give a  $(4 + \epsilon)$ -approximation [1, 5] to the maximum weighted matching. However to bound the weighted sensitivity of our algorithm, we must choose  $\epsilon$  carefully.

Formal description of our algorithm is given in Algorithm 7. It first defines subsets of edges  $E_i$ , where we assume the weight of each edge is polynomially bounded in  $n$ , so that  $1 \leq w(e) \leq n^c$  for some constant  $c$ . For a parameter  $\alpha > 1$ , we define  $E_i$  to be the subset of edges in  $E$  with weight at least  $\alpha^i$ . Algorithm 7 first draws a random permutation  $\pi$  of the edges and greedily forms a maximal matching  $M_i$  on each set  $E_i$  induced by  $\pi$ . It then greedily adds edges to a maximal matching, starting from the matching of the heaviest weight class and moving downward. That is, we initialize  $M$  to be the empty set and greedily add edges of  $M_i$  to  $M$ , starting with  $i = O(\log_\alpha n^c)$  and decrementing  $i$  after each iteration.

► **Theorem 21** ([1, 5]). *Algorithm 7 gives an  $\frac{1}{4\alpha}$ -approximation to the maximum weighted matching and uses runtime  $O(m \log_\alpha n)$  on a graph with  $m$  edges and  $n$  vertices.*

## 5.3 Sensitivity Analysis

We first require the following key structural lemma that we use to prove Theorem 12 in the full version of the paper.

► **Lemma 22.** *In expectation, the deletion of an edge  $e$  alters at most one edge in  $M_i$ , i.e., at most one edge in  $M_i$  is inserted or deleted in expectation.*

The sensitivity analysis follows from the observation that the deletion of an edge  $e$  can only affect the matchings  $E_i$  for which  $\alpha^i \leq w(e)$ . Moreover by Lemma 22, the deletion of edge  $e$  affects at most one edge in  $E_i$ , in expectation. Thus in expectation, the deletion of  $e$  affects at most two edges in  $E_{i-1}$  in expectation and inductively, the deletion of  $e$  affects at most  $2^j$

---

**Algorithm 7** Algorithm for maximum weighted matching.

---

```

1 Procedure MATCHING( $G, \epsilon, w$ )
2   Let  $C$  be a sufficiently large constant and  $\alpha > 1$  be a trade-off parameter.;
3   Let  $\pi$  be a random ordering of the edges of  $G$ ;
4   For each  $i = 0$  to  $i = C \log n$ , let  $E_i$  be the set of edges with weight at least  $\alpha^i$ ;
5   Let  $M_i$  be the greedy maximal matching of  $E_i$  induced by  $\pi$ ;
6    $M \leftarrow \emptyset$ ;
7   for  $i = C \log n$  to 0 do
8     for  $e \in M_i$  do
9       if  $e$  is not adjacent to  $M$  then
10         $M \leftarrow M \cup \{e\}$ ;
11  return  $M$ .

```

---

edges in  $E_{i-j}$  in expectation. On the other hand, the weight of each edge in  $E_{i-j}$  is at most  $\alpha^{i-j}$  so the weighted sensitivity is  $\sum_j \alpha^{i-j} 2^j$ . Hence for  $\alpha = 2$ , the weighted sensitivity is  $O(2^i \log n)$  and for  $\alpha > 2$ , the weighted sensitivity is  $O(2^i)$ . Similarly for normalized weighted sensitivity, we rescale by  $\frac{1}{2^i}$  so that the normalized weighted sensitivity is  $O(\log n)$  for  $\alpha = 2$  and  $O(1)$  for  $\alpha > 2$ . We now formalize this intuition.

► **Theorem 23.** *Suppose  $1 \leq w(e) \leq W \leq n^c$  for some absolute constants  $W, c > 0$  for all  $e \in E$ . The weighted sensitivity of Algorithm 7 is  $O(W \log n)$  for  $\alpha = 2$  and  $O(W)$  for  $\alpha > 2$ . The normalized weighted sensitivity of Algorithm 7 is  $O(\log n)$  for  $\alpha = 2$  and  $O(1)$  for  $\alpha > 2$ .*

**Proof.** Let  $e$  be an edge of weight  $w(e) \in [2^i, 2^{i+1}]$  for some integer  $i \geq 0$  and suppose  $e$  is removed from  $G$ . For  $j \leq i$ , let  $S_j$  be the set of edges in  $E_j$  affected by the deletion of edge  $e$ , so that by Lemma 22,  $\mathbb{E}[|S_i|] \leq 1$ . Then we have  $\mathbb{E}[|S_i| \cup \{e\}] \leq 2$  so that Lemma 22 implies that  $\mathbb{E}[|S_{i-1}|] \leq 2$ . Now suppose that for a fixed  $j \leq i$ , we have  $\mathbb{E}[|\{e\} \cup \bigcup_{k=j}^i S_k|] \leq 2^{i-j}$ . Then Lemma 22 implies that  $\mathbb{E}[|S_{j-1}|] \leq 2^{i-j}$  so that

$$\mathbb{E} \left[ \left| \{e\} \cup \bigcup_{k=j-1}^i S_k \right| \right] \leq 2^{i-j+1}.$$

Hence by induction, we have  $\mathbb{E}[|S_j|] \leq 2^{i-j}$ .

Since each edge of  $E_i$  has weight at most  $\alpha^i$ , then we have

$$d_{\text{EM}}(A(G), A(G - e)) = \sum_{j=0}^i 2^{i-j} \alpha^j,$$

where  $A(G)$  represents the output of Algorithm 7 on  $G$ . Under the assumption that  $1 \leq w(e) \leq n^c$  for some absolute constant  $c > 0$  for all  $e \in E$ , then  $i = O(\log n)$ . Hence for  $\alpha = 2$ , we have  $\sum_{j=0}^i 2^{i-j} \alpha^j = O(2^i \log n) = O(W \log n)$  and for  $\alpha > 2$ , we have  $\sum_{j=0}^i 2^{i-j} \alpha^j = O(2^i) = O(W)$ . Moreover, we have

$$\overline{d_{\text{EM}}}(A(G), A(G - e)) \leq \frac{1}{2^{i+1}} \sum_{j=0}^i 2^{i-j} \alpha^j,$$

so that  $\overline{d_{\text{EM}}}(A(G), A(G - e)) \leq O(\log n)$  for  $\alpha = 2$  and  $\overline{d_{\text{EM}}}(A(G), A(G - e)) = O(1)$  for  $\alpha > 2$ . ◀

Together, Theorem 21 and Theorem 23 give the full guarantees of Algorithm 7.

► **Theorem 24.** *Let  $G = (V, E)$  be a weighted graph with  $w(e) \leq W \leq n^c$  for some constant  $c > 0$  and all  $e \in E$ . For a trade-off parameter  $\alpha$ , there exists an algorithm that outputs a  $\frac{1}{4\alpha}$ -approximation to the maximum weighted matching in  $O(m \log_\alpha n)$  time. For  $\alpha = 2$ , the algorithm has weighted sensitivity  $O(W \log n)$  and normalized weighted sensitivity  $O(\log n)$ . For  $\alpha > 2$ , the algorithm has weighted sensitivity  $O(W)$  and normalized weighted sensitivity  $O(1)$ .*

We again emphasize that the worst case weighted sensitivity of *any* constant factor approximation algorithm to the maximum weighted matching problem is at least  $\Omega(W)$ . Recall that if an edge of weight  $W = n^c$  is altered in a graph whose remaining edges have weight 1, then any constant factor approximation to the maximum weighted matching must include the heavy edge for sufficiently large  $c > 0$ , which incurs cost  $\Omega(W)$  in the weighted sensitivity. Thus for  $\alpha > 2$ , Algorithm 7 performs well with respect to both weighted sensitivity and normalized weighted sensitivity.

---

## References

- 1 Marc Bury, Elena Grigorescu, Andrew McGregor, Morteza Monemizadeh, Chris Schwiegelshohn, Sofya Vorotnikova, and Samson Zhou. Structural results on matching estimation with applications to streaming. *Algorithmica*, 81(1):367–392, 2019.
- 2 Keren Censor-Hillel, Elad Haramaty, and Zohar Karnin. Optimal dynamic distributed mis. In *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing (PODC)*, pages 217–226, 2016.
- 3 Vincent Cohen-Addad, Niklas Hjuler, Nikos Parotsidis, David Saulpic, and Chris Schwiegelshohn. Fully dynamic consistent facility location. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 3250–3260, 2019.
- 4 Richard Cole and Uzi Vishkin. Deterministic coin tossing with applications to optimal parallel list ranking. *Information and Control*, 70(1):32–53, 1986.
- 5 Michael Crouch and Daniel S. Stubbs. Improved streaming algorithms for weighted matching, via unweighted matching. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*, volume 28, pages 96–104, 2014.
- 6 P. Erdős and R. Rado. Combinatorial theorems on classifications of subsets of a given set. *Proceedings of the London Mathematical Society*, s3-2(1):417–439, 1952.
- 7 Guy Even, Moti Medina, and Dana Ron. Distributed maximum matching in bounded degree graphs. In *Proceedings of the International Conference on Distributed Computing and Networking (ICDCN)*, pages 18:1–18:10, 2015.
- 8 Silvio Lattanzi and Sergei Vassilvitskii. Consistent  $k$ -clustering. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 1975–1984, 2017.
- 9 Andrew McGregor. Finding graph matchings in data streams. In *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*, pages 170–181, 2005.
- 10 Michal Parnas and Dana Ron. Approximating the minimum vertex cover in sublinear time and a connection to distributed algorithms. *Theoretical Computer Science*, 381(1-3):183–196, 2007.
- 11 Pan Peng and Yuichi Yoshida. Average sensitivity of spectral clustering. In *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 1132–1140, 2020.
- 12 Nithin Varma and Yuichi Yoshida. Average sensitivity of graph algorithms. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2021. to appear.

# Computational Complexity of the Hylland-Zeckhauser Scheme for One-Sided Matching Markets

Vijay V. Vazirani

Department of Computer Science, University of California, Irvine, CA, USA  
vazirani@ics.uci.edu

Mihalis Yannakakis

Department of Computer Science, Columbia University, New York, NY, USA  
mihalis@cs.columbia.edu

---

## Abstract

In 1979, Hylland and Zeckhauser [23] gave a simple and general scheme for implementing a one-sided matching market using the power of a pricing mechanism. Their method has nice properties – it is incentive compatible in the large and produces an allocation that is Pareto optimal – and hence it provides an attractive, off-the-shelf method for running an application involving such a market. With matching markets becoming ever more prevalent and impactful, it is imperative to finally settle the computational complexity of this scheme.

We present the following partial resolution:

1. A combinatorial, strongly polynomial time algorithm for the dichotomous case, i.e., 0/1 utilities, and more generally, when each agent’s utilities come from a bi-valued set.
2. An example that has only irrational equilibria, hence proving that this problem is not in PPAD.
3. A proof of membership of the problem in the class FIXP.
4. A proof of membership of the problem of computing an approximate HZ equilibrium in the class PPAD.

We leave open the (difficult) questions of determining if computing an exact HZ equilibrium is FIXP-hard and an approximate HZ equilibrium is PPAD-hard.

**2012 ACM Subject Classification** Theory of computation → Algorithmic game theory

**Keywords and phrases** Hylland-Zeckhauser scheme, one-sided matching markets, mechanism design, dichotomous utilities, PPAD, FIXP

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.59

**Related Version** Full version: <https://arxiv.org/pdf/2004.01348>

**Funding** *Vijay V. Vazirani*: Supported in part by NSF grant CCF-1815901.

*Mihalis Yannakakis*: Supported in part by NSF grants CCF-1703925 and CCF-1763970.

**Acknowledgements** We wish to thank Federico Echenique, Jugal Garg, Tung Mai and Thorben Trobst for valuable discussions and Richard Zeckhauser for providing us with the Appendix to his paper [23]. In addition, the first author wishes to thank Simons Institute for running a program on matching markets in Fall 2019; this provided valuable exposure to the topic.

## 1 Introduction

In a brilliant and by-now classic paper, Hylland and Zeckhauser [23] gave a simple and general scheme for implementing a one-sided matching market using the power of a pricing mechanism<sup>1</sup>. Their method produces an allocation that is Pareto optimal, envy-free [23]

---

<sup>1</sup> See Remark 5 for a discussion of the advantages of this mechanism.



and is incentive compatible in the large [22]. The Hylland-Zeckhauser (HZ) scheme can be viewed as a marriage between fractional perfect matching and a linear Fisher market, both of which admit not only polynomial time algorithms but also combinatorial ones. These facts have enticed numerous researchers over the years to seek an efficient algorithm for the HZ scheme. The significance of this problem has only grown in recent years, with ever more diverse and impactful matching markets being launched into our economy, e.g., see [17].

Our work on resolving this problem started with an encouraging sign, when we obtained a combinatorial, strongly polynomial time algorithm for the *dichotomous case*, in which all utilities are 0/1, by melding a perfect matching algorithm with the combinatorial algorithm of [13] for the linear Fisher market, see Section 4. This algorithm can be extended to solve a more general problem which we call the *bi-valued utilities case*, in which each agent's utilities can take one of only two values, though the two values can be different for different agents. However, this approach did not extend any further, as described in the next section.

One-sided matching markets can be classified along two dimensions: whether the utility functions are cardinal or ordinal, and whether agents have initial endowments or not. Under this classification, the HZ scheme is (cardinal, no endowments). Section 1.2 gives mechanisms for the remaining three possibilities as well as their game-theoretic properties. Ordinal and cardinal utility functions have their individual pros and cons, and neither dominates the other. Whereas the former are easier to elicit, the latter are far more expressive, enabling an agent to not only report if she prefers one good to another but also by how much, thereby producing higher quality allocations as illustrated in Example 1, which is taken from [20].

► **Example 1.** ([20]) The instance has three types of goods,  $T_1, T_2, T_3$ , and these goods are present in the proportion of (1%, 97%, 2%). Based on their utility functions, the agents are partitioned into two sets  $A_1$  and  $A_2$ , where  $A_1$  constitute 1% of the agents and  $A_2$ , 99%. The utility functions of agents in  $A_1$  and  $A_2$  for the three types of goods are  $(1, \epsilon, 0)$  and  $(1, 1 - \epsilon, 0)$ , respectively, for a small number  $\epsilon > 0$ . The main point is that whereas agents in  $A_2$  marginally prefer  $T_1$  to  $T_2$ , those in  $A_1$  overwhelmingly prefer  $T_1$  to  $T_2$ . Clearly, the ordinal utilities of all agents in  $A_1 \cup A_2$  are the same. Therefore, a mechanism based on such utilities will not be able to make a distinction between the two types of agents. On the other hand, the HZ mechanism, which uses cardinal utilities, will fix the price of goods in  $T_3$  to be zero and those in  $T_1$  and  $T_2$  appropriately so that by-and-large the bundles of  $A_1$  and  $A_2$  consist of goods from  $T_1$  and  $T_2$ , respectively.

While studying the dichotomous case of two-sided markets, Bogomolnaia and Moulin [5] called it an “important special case of the bilateral matching problem.” Using the Gallai-Edmonds decomposition of a bipartite graph, they gave a mechanism that is Pareto optimal and group strategyproof. They also gave a number of applications of their setting, some of which are natural applications of one-sided markets as well, e.g., housemates distributing rooms, having different features, in a house. Furthermore, they say, “Time sharing is the simplest way to deal fairly with indivisibilities of matching markets: think of a set of workers sharing their time among a set of employers.” It turns out that the HZ (fractional) equilibrium allocation is a superior starting point for the problem of designing a randomized time-sharing mechanism; this is discussed in Remark 5 after introducing the HZ model. Roth, Sonmez and Unver [29] extended these results to general graph matching under dichotomous utilities; this setting is applicable to the kidney exchange marketplace.



## 1.1 The gamut of possibilities

The most useful solution for practical applications would of course have been a combinatorial, polynomial time algorithm for the entire scheme. At the outset, this didn't seem unlikely, especially in view of the existence of such an algorithm for the dichotomous case. Next we considered the generalization of the bi-valued utilities case to tri-valued utilities, in particular, to the case of  $\{0, \frac{1}{2}, 1\}$  utilities. However, even this case appears to be intractable.

Underlying the polynomial time solvability of a linear Fisher market is the property of weak gross substitutability<sup>2</sup>. We note that this property is destroyed as soon as one goes to a slightly more general utility function, namely piecewise-linear, concave and separable over goods (SPLC utilities), and this case is PPAD-complete<sup>3</sup> [32]; the class PPAD was introduced in [28]. Since equilibrium allocations for the HZ scheme do not satisfy weak gross substitutability, e.g., see Example 9, we were led us to seek a proof of PPAD-completeness.

A crucial requirement for membership in PPAD is to show that there is always a rational equilibrium if all parameters of the instance are rational numbers. However, even this is not true; we found an example which admits only irrational equilibria, see Section 6. This example consists of four agents and goods, and hence can be viewed as belonging to the four-valued utilities case; see Remark 17 for other intriguing aspects of this example.

The irrationality of solutions suggests that the appropriate class for this problem is the class FIXP, introduced in [16]. The proof in [23], showing the existence of an equilibrium, uses Kakutani's theorem and does not seem to lend itself in any easy way to showing membership in FIXP. For this purpose, we give a new proof of the existence of equilibrium. Our proof defines a suitable Brouwer function which adjusts prices and allocations in case they are not an equilibrium. It uses elementary arithmetic operations that improve their optimality or feasibility of the current prices and allocations. The adjustment scheme is such that the only stable prices and allocations are forced to be equilibria.

Next, we define the notion of an approximate equilibrium. This is still required to be a fractional perfect matching on agents and goods; agents' allocations are allowed to be slightly suboptimal and/or their cost is allowed to slightly exceed the budget of 1 dollar. We show that the problem of computing such an approximate equilibrium is in PPAD. This involves relating approximate equilibria to the approximate fixed points of the Brouwer function we defined for our proof of membership in FIXP. We leave open the questions of determining if the computation (to desired accuracy) of an exact HZ equilibrium is FIXP-hard, and if the computation of an approximate HZ equilibrium is PPAD-hard.

## 1.2 Related work

We first present mechanisms for the remaining three possibilities for the classification of one-sided matching market mechanisms given in the Introduction. The famous Top Trading Cycles mechanism is (ordinal, endowments) [30]; it is efficient, strategyproof and core-stable. Under (ordinal, no endowments) are Random Priority [27], which is strategyproof though not efficient or envy-free, and Probabilistic Serial [4], which is efficient and envy-free but not strategyproof. Under (cardinal, endowments) is  $\epsilon$ -Approximate ADHZ (for Arrow-Debreu Hylland-Zeckhauser) scheme [20], which satisfies Pareto optimality, approximate envy-freeness and incentive compatibility in the large.

<sup>2</sup> Namely, if you increase the price of one good, the demand of another good cannot decrease.

<sup>3</sup> Independently, PPAD-hardness was also established in [10].

We are aware of only the following two computational results on the HZ scheme. Using the algebraic cell decomposition technique of [2], [12] gave a polynomial time algorithm for computing an equilibrium for an Arrow-Debreu market under piecewise-linear, concave (PLC) utilities (not necessarily separable over goods) if the number of goods is fixed. One can see that their algorithm can be adapted to yield a polynomial time algorithm for computing an equilibrium for the HZ scheme if the number of goods is a fixed constant. Extending these methods, [1] gave a polynomial time algorithm for the case that the number of agents is a fixed constant.

There are several results establishing membership and hardness in PPAD and FIXP for equilibria computation problems in different settings. The quintessential complete problem for PPAD is 2-Nash [11, 7] and that for FIXP is multiplayer Nash equilibrium [16]. For the latter problem, computing an approximate equilibrium is PPAD-complete [11].

For the case of market equilibria, in the economics literature, there are two parallel streams of results: one assumes that an excess demand function is given and the other assumes a specific class of utility functions. [16] proved FIXP-completeness of Arrow-Debreu markets whose excess demand functions are algebraic. This result is for the first stream and it does not establish FIXP-completeness of Arrow-Debreu markets under any specific class of utility functions. Results for the second stream include proofs of membership in FIXP for Arrow-Debreu markets under Leontief and piecewise-linear concave (PLC) utility functions in [35] and [18], respectively. This was followed by a proof of FIXP-hardness for Arrow-Debreu markets with Leontief and PLC utilities [19]. For the case of Arrow-Debreu markets with CES (constant elasticity of substitution) utility functions, [9] show membership in FIXP but leave open FIXP-hardness.

For the CES market problem stated above, computing an approximate equilibrium is PPAD-complete, and the same holds more generally for a large class of “non-monotonic” markets [9]. Computing an (exact or approximate) equilibrium under separable, piecewise-linear, concave (SPLC) utilities for Arrow-Debreu and Fisher markets is also known to be PPAD-complete [8, 10, 32].

In recent years, several researchers have proposed Hylland-Zeckhauser-type mechanisms for a number of applications, e.g., see [6, 22, 25, 26]. The basic scheme has also been generalized in several different directions, including two-sided matching markets, adding quantitative constraints, and to the setting in which agents have initial endowments of goods instead of money, see [14, 15].

## 2 The Hylland-Zeckhauser Scheme

Hylland and Zeckhauser [23] gave a general mechanism for a one-sided matching market using the power of a pricing mechanism. Their formulation is as follows: Let  $A = \{1, 2, \dots, n\}$  be a set of  $n$  agents and  $G = \{1, 2, \dots, n\}$  be a set of  $n$  indivisible goods. The mechanism will allocate exactly one good to each agent and will have the following two properties:

- The allocation produced is Pareto optimal.
- The mechanism is incentive compatible in the large.

The Hylland-Zeckhauser scheme is a marriage between linear Fisher market and fractional perfect matching. The agents will reveal to the mechanism their desires for the goods by stating their von Neumann-Morgenstern utilities. Let  $u_{ij}$  represent the utility of agent  $i$  for good  $j$ . We will use language from the study of market equilibria to describe the rest of the formulation. For this purpose, we next define the linear Fisher market model.

A *linear Fisher market* consists of a set  $A = \{1, 2, \dots, n\}$  of  $n$  agents and a set  $G = \{1, 2, \dots, m\}$  of  $m$  infinitely divisible goods. By fixing the units for each good, we may assume without loss of generality that there is a unit of each good in the market. Each agent  $i$  has money  $m_i$  and utility  $u_{ij}$  for a unit of good  $j$ . If  $x_{ij}$ ,  $1 \leq j \leq m$  is the *bundle of goods allocated to  $i$* , then the utility accrued by  $i$  is  $\sum_j u_{ij}x_{ij}$ . Each good  $j$  is assigned a non-negative price,  $p_j$ . Allocations and prices,  $x$  and  $p$ , are said to form an *equilibrium* if each agent obtains a utility maximizing bundle of goods at prices  $p$  and the *market clears*, i.e., each good is fully sold and all money of agents is fully spent.

In order to mold the one-sided market into a linear Fisher market, the HZ scheme renders goods divisible by assuming that there is one unit of probability share of each good. An *allocation* to an agent is a collection of probability shares over the goods. Let  $x_{ij}$  be the probability share that agent  $i$  receives of good  $j$ . Then,  $\sum_j u_{ij}x_{ij}$  is the *expected utility* accrued by agent  $i$ . Each good  $j$  has price  $p_j \geq 0$  in this market and each agent has 1 dollar with which it buys probability shares. The entire allocation must form a *fractional perfect matching in the complete bipartite graph* over vertex sets  $A$  and  $G$  as follows: there is one unit of probability share of each good and the total probability share assigned to each agent also needs to be one unit. Subject to these constraints, each agent should buy a utility maximizing bundle of goods *having the smallest possible cost*. Note that the last condition is not required in the definition of a linear Fisher market equilibrium. It is needed here to guarantee that the allocation obtained is Pareto optimal, see [23] for proof of Pareto optimality. A second departure from the linear Fisher market equilibrium is that in the latter, each agent  $i$  must spend her money  $m_i$  fully; in the HZ scheme,  $i$  need not spend her entire dollar. Since the allocation is required to form a fractional perfect matching, all goods are fully sold. We will define these to be *equilibrium allocation and prices*; we state this formally below after giving some preliminary definitions.

► **Definition 2.** Let  $x$  and  $p$  denote arbitrary (non-negative) allocations and prices of goods. By *size*, *cost* and *value* of agent  $i$ 's bundle we mean

$$\sum_{j \in G} x_{ij}, \quad \sum_{j \in G} p_j x_{ij} \quad \text{and} \quad \sum_{j \in G} u_{ij} x_{ij},$$

respectively. We will denote these by  $\text{size}(i)$ ,  $\text{cost}(i)$  and  $\text{value}(i)$ , respectively.

► **Definition 3** (Hylland and Zeckhauser [23]). *Allocations and prices  $(x, p)$  form an equilibrium for the one-sided matching market stated above if:*

1. *The total probability share of each good  $j$  is 1 unit, i.e.,  $\sum_i x_{ij} = 1$ .*
2. *The size of each agent  $i$ 's allocation is 1, i.e.,  $\text{size}(i) = 1$ .*
3. *The cost of the bundle of each agent is at most 1.*
4. *Subject to constraints 2 and 3, each agent  $i$  maximizes her expected utility at minimum possible cost, i.e., maximize  $\text{value}(i)$ , subject to  $\text{size}(i) = 1$ ,  $\text{cost}(i) \leq 1$ , and lastly,  $\text{cost}(i)$  is smallest among all utility-maximizing bundles of  $i$ .*

Using Kakutani's fixed point theorem, the following is shown:

► **Theorem 4** (Hylland and Zeckhauser [23]). *Every instance of the one-sided market defined above admits an equilibrium; moreover, the corresponding allocation is Pareto optimal.*

Finally, if this “market” is large enough, no individual agent will be able to improve her allocation by misreporting utilities nor will she be able to manipulate prices. For this reason, the HZ scheme is incentive compatible in the large.

As stated above, Hylland and Zeckhauser view each agent's allocation as a lottery over goods. In this viewpoint, agents accrue utility in an *expected sense* from their allocations. Once these lotteries are resolved in a manner faithful to the probabilities, an assignment of indivisible goods will result. The latter can be done using the well-known Theorem of Birkhoff [3] and von Neumann [34] which states that any doubly stochastic matrix can be written as a convex combination of permutation matrices, i.e., perfect matchings; moreover, this decomposition can be obtained efficiently. Next, pick one of these perfect matchings from the discrete distribution given by coefficients in the convex combination. As is well known, since the lottery over goods is Pareto optimal *ex ante*, the integral allocation, viewed stochastically, will also be Pareto optimal *ex post*.

Another viewpoint, forwarded by Bogomolnaia and Moulin [5], considers the fractional perfect matching, or equivalently the doubly-stochastic matrix, as the output of the mechanism, i.e., without resorting to randomized rounding. This viewpoint assumes that the agents are going to “time-share” the goods or resources and the doubly-stochastic matrix, which is derived from a market mechanism, provides a “fair” way of doing so.

► **Remark 5.** In their paper studying the dichotomous case of two-sided matching markets, Bogomolnaia and Moulin [5] state that the preferred way of dealing with indivisibilities inherent in matching markets is to resort to time sharing using randomization. Their method builds on the Gallai-Edmonds decomposition of the underlying bipartite graph; this classifies vertices into three categories: disposable, over-demanded and perfectly matched. This is a much more coarse insight into the demand structure of vertices than that obtained via the HZ equilibrium. The latter is the output of a market mechanism in which equilibrium prices reflect the relative importance of goods in an accurate and precise manner, based on the utilities declared by buyers, and equilibrium allocations are as equitable as possible across buyers. Hence the latter yields a more fair and desirable randomized time-sharing mechanism.

### 3 Properties of Optimal Allocations and Prices

Let  $p$  be given prices which are not necessarily equilibrium prices. An optimal bundle for agent  $i$ ,  $x_i$ , is a solution to the following LP, which has two constraints, one for size and one for cost.

$$\max \sum_j x_{ij} u_{ij} \quad (1)$$

$$\text{s.t.} \quad (2)$$

$$\sum_j x_{ij} = 1 \quad (3)$$

$$\sum_j x_{ij} p_j \leq 1 \quad (4)$$

$$\forall j \quad x_{ij} \geq 0 \quad (5)$$

Taking  $\mu_i$  and  $\alpha_i$  to be the dual variables corresponding to the two constraints, we get the dual LP:

$$\min \quad \alpha_i + \mu_i \quad (6)$$

$$\text{s.t.} \quad (7)$$

$$\forall i, j \quad \alpha_i p_j + \mu_i \geq u_{ij} \quad (8)$$

$$\alpha_i \geq 0 \quad (9)$$

Clearly  $\mu_i$  is unconstrained.  $\mu_i$  will be called the *offset* on  $i$ 's utilities. By complementary slackness, if  $x_{ij}$  is positive then  $\alpha_i p_j = u_{ij} - \mu_i$ . All goods  $j$  satisfying this equality will be called *optimal goods for agent  $i$* . The rest of the goods, called *suboptimal*, will satisfy  $\alpha_i p_j > u_{ij} - \mu_i$ . Obviously an optimal bundle for  $i$  must contain only optimal goods.

The parameter  $\mu_i$  plays a crucial role in ensuring that  $i$ 's optimal bundle satisfies both size and cost constraints. If a single good is an effective way of satisfying both size and cost constraints, then  $\mu_i$  plays no role and can be set to zero. However, if different goods are better from the viewpoint of size and cost, then  $\mu_i$  attains the right value so they both become optimal and  $i$  buys an appropriate combination. We provide an example below to illustrate this.

► **Example 6.** Suppose  $i$  has positive utilities for only two goods,  $j$  and  $k$ , with  $u_{ij} = 10$ ,  $u_{ik} = 2$  and their prices are  $p_j = 2$ ,  $p_k = 0.1$ . Clearly, neither good satisfies both size and cost constraints optimally: good  $j$  is better for the size constraint and  $k$  is better for the cost constraint. If  $i$  buys one unit of good  $j$ , she spends 2 dollars, thus exceeding her budget. On the other hand, she can afford to buy 10 units of  $k$ , giving her utility of 20; however, she has far exceeded the size constraint. It turns out that her optimal bundle consists of 9/19 units of  $j$  and 10/19 units of  $k$ ; the costs of these two goods being 18/19 and 1/19 dollars, respectively. Clearly, her size and cost constraints are both met exactly. Her total utility from this bundle is 110/19. It is easy to see that  $\alpha_i = 80/19$  and  $\mu_i = 30/19$ , and for these settings of the parameters, both goods are optimal.

We next show that equilibrium prices are invariant under the operation of *scaling* the difference of prices from 1.

► **Lemma 7.** *Let  $p$  be an equilibrium price vector and fix any  $r > 0$ . Let  $p'$  be such that  $\forall j \in G, p'_j - 1 = r(p_j - 1)$ . Then  $p'$  is also an equilibrium price vector.*

**Proof.** Consider an agent  $i$ . Clearly,  $\sum_{j \in G} p_j x_{ij} \leq 1$ . Now,

$$\sum_{j \in G} p'_j x_{ij} = \sum_{j \in G} (rp_j - r + 1)x_{ij} \leq 1,$$

where the last inequality follows by using  $\sum_{j \in G} x_{ij} = 1$ . ◀

Using Lemma 7, it is easy to see that if the allocation  $x$  provides optimal bundles to all agents under prices  $p$  then it also does so under  $p'$ . In the rest of this paper we will enforce that the minimum price of a good is zero, thereby fixing the scale. Observe that the main goal of the Hylland-Zeckhauser scheme is to yield the “correct” allocations to agents; the prices are simply a vehicle in the market mechanism to achieve this. Hence arbitrarily fixing the scale does not change the essential nature of the problem. Moreover, setting the minimum price to zero is standard [23] and can lead to simplifying the equilibrium computation problem as shown in Remark 8.

► **Remark 8.** We remark that on the one hand, the offset  $\mu_i$  is a key enabler in construing optimal bundles, on the other, it is also a main source of difficulty in computing equilibria for the HZ scheme. We identify here an interesting case in which  $\mu_i = 0$  and this difficulty is mitigated. In particular, this holds for all agents in the dichotomous case presented in Section 4. Suppose good  $j$  is optimal for agent  $i$ ,  $u_{ij} = 0$  and  $p_j = 0$ , then it is easy to check that  $\mu_i = 0$ . If so, the optimal goods for  $i$  are simply the maximum bang-per-buck goods; the latter notion is replete in market equilibrium papers, e.g., see [13].

Finally, we extend Example 6 to illustrate that optimal allocations for the Hylland-Zeckhauser model do not satisfy the weak gross substitutes condition in general.

► **Example 9.** In Example 6, let us raise the price of  $k$  to 0.2 dollars. Then the optimal allocation for  $i$  changes to  $4/9$  units of  $j$  and  $5/9$  units of  $k$ . Notice that the demand for  $j$  went down from  $9/19$  to  $4/9$ . One way to understand this change is as follows: Let us start with the old allocation of  $10/19$  units of  $k$ . Clearly, the cost of this allocation of  $k$  went up from  $1/19$  to  $2/19$ , leaving only  $17/19$  dollars for  $j$ . Therefore size of  $j$  needs to be reduced to  $17/38$ . However, now the sum of the sizes becomes  $37/38$ , i.e., less than a unit. We wish to increase this to a unit while still keeping cost at a unit. The only way of doing this is to sell some of the more expensive good and use the money to buy the cheaper good. This is the reason for the decrease in demand of  $j$ .

#### 4 Strongly Polynomial Algorithm for Bi-Valued Utilities

In this section, we will study the restriction of the HZ scheme to the bi-valued utilities case, which is defined as follows: for each agent  $i$ , we are given a set  $\{a_i, b_i\}$ , where  $0 \leq a_i < b_i$ , and the utilities  $u_{ij}$ ,  $\forall j \in G$ , are picked from this set. However first, using a perfect matching algorithm and the combinatorial algorithm [13] for linear Fisher markets, we will give a strongly polynomial time algorithm for the dichotomous case, i.e., when all utilities  $u_{ij}$  are 0/1. Next we define the notion of equivalence of utility functions and show that the bi-valued utilities case is equivalent to the dichotomous case, thereby extending the dichotomous case algorithm to this case.

We need to clarify that we will not use the main algorithm from [13], which uses the notion of balanced flows and  $l_2$  norm to achieve polynomial running time. Instead, we will use the “simple algorithm” presented in Section 5 in [13]. Although this algorithm is not proven to be efficient, the simplified version we define below, called Simplified DPSV Algorithm, is efficient; in fact it runs in strongly polynomial time, unlike the balanced-flows-based algorithm of [13]. Remark 8 provides an insight into what makes the dichotomous case computationally easier.

We note that recently, [20] gave a *rational convex program (RCP)* for the dichotomous case of HZ, and more recently, [33] made a small modification to our algorithm to obtain a mechanism that is proven to be strategyproof. An RCP, defined in [31], is a nonlinear convex program all of whose parameters are rational numbers and which always admits a rational solution in which the denominators are polynomially bounded. An RCP can be solved exactly in polynomial time using the ellipsoid algorithm and diophantine approximation [21, 24], and therefore directly implies the existence of a polynomial time algorithm for the underlying problem.

**Notation.** We will denote by  $H = (A, G, E)$  be the bipartite graph on vertex sets  $A$  and  $G$ , and edge set  $E$ , with  $(i, j) \in E$  iff  $u_{ij} = 1$ . For  $A' \subseteq A$  and  $G' \subseteq G$ , we will denote by  $H[A', G']$  the restriction of  $H$  to vertex set  $A' \cup G'$ . If  $\nu$  is a matching in  $H$ ,  $\nu \subseteq E$ , and  $(i, j) \in \nu$  then we will say that  $\nu(i) = j$  and  $\nu(j) = i$ . For any subset  $S \subseteq A$  ( $S \subseteq G$ ),  $N(S)$  will denote the set of neighbors, in  $G$  ( $A$ ), of vertices in  $S$ .

If  $H$  has a perfect matching, the matter is straightforward as stated in Steps 1a and 1b; allocations and prices are clearly in equilibrium. For Step 2, we need the following lemma.

► **Lemma 10.** *The following hold:*

1. For any set  $S \subseteq A_2$ ,  $|N(S)| \geq |S|$ .
2. For any set  $S \subseteq G_1$ ,  $|N(S) \cap A_1| \geq |S|$ .

**Proof.**

1. If  $|N(S)| < |S|$  then  $(G_1 \cup N(S)) \cup (A_2 - S)$  is a smaller vertex cover for  $H$ , leading to a contradiction.
2. If  $|N(S) \cap A_1| < |S|$  then  $(G_1 - S) \cup (A_2 \cup N(S))$  is a smaller vertex cover for  $H$ , leading to a contradiction.  $\blacktriangleleft$

The first part of Lemma 10 together with Hall's Theorem implies that a maximum matching in  $H[A_2, G_2]$  must match all agents. Therefore in Step 2a, each agent  $i \in A_2$  is allocated one unit of a unique good from which it derives utility 1 and having price zero; clearly, this is an optimal bundle of minimum cost for  $i$ . The number of goods that will remain unmatched in  $G_2$  at the end of this step is  $|G_2| - |A_2|$ .

■ **Algorithm 1** Algorithm for the Dichotomous Case.

---

1. If  $H$  has a perfect matching, say  $\nu$ , then do:
    - a.  $\forall i \in A$ : allocate good  $\nu(i)$  to  $i$ .
    - b.  $\forall j \in G$ :  $p_j \leftarrow 0$ . Go to Step 3.
  2. Else do:
    - a. Find a minimum vertex cover in  $H$ , say  $G_1 \cup A_2$ , where  $G_1 \subset G$  and  $A_2 \subset A$ .  
Let  $A_1 = A - A_2$  and  $G_2 = G - G_1$ .
    - b. Find a maximum matching in  $H[A_2, G_2]$ , say  $\nu$ .
    - c.  $\forall i \in A_2$ : allocate good  $\nu(i)$  to  $i$ .
    - d.  $\forall j \in G_2$ :  $p_j \leftarrow 0$ .
    - e. Run the Simplified DPSV Algorithm on agents  $A_1$  and goods  $G_1$ .
    - f.  $\forall i \in A_1$ : Allocate unmatched goods of  $G_2$  to satisfy the size constraint.
  3. Output the allocations and prices computed and Halt.
- 

Allocations are computed for agents in  $A_1$  as follows. First, Step 2e uses the Simplified DPSV Algorithm, which we describe below, to compute equilibrium allocations and prices for the submarket consisting of agents in  $A_1$  and goods in  $G_1$ . At the end of this step, the money of each agent in  $A_1$  is exhausted; however, her allocation may not meet the size constraint. To achieve the latter, Step 2f allocates the unmatched zero-priced goods from  $G_2$  to agents in  $A_1$ . Clearly, the total deficit in size is  $|A_1| - |G_1|$ . Since this equals  $|G_2| - |A_2|$ , the market clears at the end of Step 2f. As shown in Lemma 11, each agent in  $A_1$  also gets an optimal bundle of goods of minimum cost.

Let  $p$  be the prices of goods in  $G_1$  at any point in this algorithm. As a consequence of the second part of Lemma 10, the equilibrium price of each good in  $G_1$  will be at least 1. The Simplified DPSV algorithm will initialize prices of goods in  $G_1$  to 1 and declare all goods active. The algorithm will always raise prices of active goods uniformly<sup>4</sup>.

For  $S \subseteq G_1$  let  $p(S)$  denote the sum of the equilibrium prices of goods in  $S$ . A key notion from [13] is that of a tight set; set  $S \subseteq G_1$  is said to be *tight* if  $p(S) = |N(S)|$ , the latter being the total money of agents in  $A_1$  who are interested in goods in  $S$ . If set  $S$  is tight, then the local market consisting of goods in  $S$  and agents in  $N(S)$  clears. To see this, one needs to use the flow-based procedure given in [13] to show that each agent  $i \in N(S)$  can be allocated 1 dollar worth of those goods in  $S$  from which it accrues unit utility. Thus equilibrium has been reached for goods in  $S$ .

---

<sup>4</sup> In [13], prices of active goods are raised multiplicatively, which amounts to raising prices of active goods uniformly for our simplified setting.



As the algorithm raises prices of all goods in  $G_1$ , at some point a set  $S$  will go tight. The algorithm then *freezes* the prices of its goods and removes them from the active set. It then proceeds to raise the prices of currently active goods until another set goes tight, and so on, until all goods in  $G_1$  are frozen.

We can now explain in what sense we need a “simplified” version of the DPSV algorithm. Assume that at some point,  $S \subset G_1$  is frozen and goods in  $G_1 - S$  are active and their prices are raised. As this happens, agents in  $A_1 - N(S)$  start preferring goods in  $S$  relative to those in  $G_1 - S$ . In the general case, at some point, an agent  $i \in (A_1 - N(S))$  will prefer a good  $j \in S$  as much as her other preferred goods. At this point, edge  $(i, j)$  is added to the active graph. As a result, some set  $S' \subseteq S$ , containing  $j$ , will not be tight anymore and will be unfrozen. However, in our setting, the utilities of agents in  $(A_1 - N(S))$  for goods in  $S$  is zero, and therefore no new edges are introduced and tight sets never become unfrozen. Hence the only events of the Simplified DPSV Algorithm are raising of prices and freezing of sets. Clearly, there will be at most  $n$  freezings. One can check details in [13] to see that the steps executed with each freezing run in strongly polynomial time, hence making the Simplified DPSV Algorithm a strongly polynomial time algorithm<sup>5</sup>.

► **Lemma 11.** *Each agent in  $A_1$  will get an optimal bundle of goods of minimum cost.*

**Proof.** First note that for agents in  $A_1$ , there are no utility 1 goods in  $G_2$  – this follows from the fact that no vertices from  $A_1 \cup G_2$  are in the vertex cover picked. Therefore, for  $i \in A_1$ , an optimum bundle consists of the cheapest way of obtaining one dollar worth of goods from  $N\{i\}$ , which are in  $G_1$ , together with the right amount of zero-priced goods from  $G_2$  to satisfy the size constraint.

Assume that the algorithm freezes  $k$  sets,  $S_1, \dots, S_k$ , in that order; the union of these sets being  $G_1$ . Let  $p_1, p_2, \dots, p_k$  be the prices of goods in these sets, respectively. Clearly, successive freezings will be at higher and higher prices and therefore,  $1 \leq p_1 < p_2 < \dots < p_k$ , and for  $1 \leq j \leq k$ ,  $p_j = |N(S_j)|/|S_j|$ . If  $i \in N(S_j)$ , the algorithm will allocate  $1/p_j$  amount of goods to  $i$  from  $S_j$ , costing 1 dollar.

By definition of neighborhood of sets, if  $i \in N(S_j)$ , then  $i$  cannot have edges to  $S_1, \dots, S_{j-1}$  and can have edges to  $S_{j+1}, \dots, S_k$ . Therefore, the cheapest goods from which it accrues unit utility are in  $S_j$ , the set from which she gets 1 dollar worth of allocation. The rest of the allocation of  $i$ , in order to meet  $i$ 's size constraint, will be from  $G_2$ , which are zero-priced and from which  $i$  gets zero utility. Clearly,  $i$  gets an optimal bundle of minimum cost. ◀

Since all steps of the algorithm, namely finding a maximum matching, a minimum vertex cover and running the Simplified DPSV Algorithm, can be executed in strongly polynomial time, we get:

► **Lemma 12.** *The algorithm given finds equilibrium prices and allocations for the dichotomous case of the Hylland-Zeckhauser scheme. It runs in strongly polynomial time.*

► **Definition 13.** *Let  $I$  be an instance of the HZ scheme and let the utility function of agent  $i$  be  $u_i = \{u_{i1}, u_{i12}, \dots, u_{in}\}$ . Then  $u'_i = \{u'_{i1}, u'_{i12}, \dots, u'_{in}\}$  is equivalent to  $u_i$  if there are two numbers  $s > 0$  and  $h \geq 0$  such that for  $1 \leq j \leq n$ ,  $u'_{ij} = s \cdot u_{ij} + h$ . The numbers  $s$  and  $h$  will be called the scaling factor and shift, respectively.*

<sup>5</sup> In contrast, in the general case, the number of freezings is not known to be bounded by a polynomial in  $n$ , as stated in [13].

► **Lemma 14.** *Let  $I$  be an instance of the HZ scheme and let the utility function of agent  $i$  be  $u_i$ . Let  $u'_i$  be equivalent to  $u_i$  and let  $I'$  be the instance obtained by replacing  $u_i$  by  $u'_i$  in  $I$ . Then  $x$  and  $p$  are equilibrium allocation and prices for  $I$  if and only if they are also for  $I'$ .*

**Proof.** Let  $s$  and  $h$  be the scaling factor and shift that transform  $u_i$  to  $u'_i$ . By the statement of the lemma,  $x_i = \{x_{i1}, \dots, x_{in}\}$  is an optimal bundle for  $i$  at prices  $p$  and hence is a solution to the optimal bundle LP (1). The objective function of this LP is

$$\sum_{j=1}^n u_{ij} x_{ij}.$$

Next observe that the objective function of the corresponding LP for  $i$  under instance  $I'$  is

$$\sum_{j=1}^n u'_{ij} x_{ij} = \sum_{j=1}^n (s \cdot u_{ij} + h) x_{ij} = h + s \cdot \sum_{j=1}^n u_{ij} x_{ij},$$

where the last equality follows from the fact that  $\sum_{j=1}^n x_{ij} = 1$ . Therefore, the objective function of the second LP is obtained from the first by scaling and shifting. Furthermore, since the constraints of the two LPs are identical, the optimal solutions of the two LPs are the same. Finally, for each  $i \in A$ : the bundle under allocation  $x$  is a minimum cost optimal bundle for  $I$  if and only if it is also for  $I'$ . The lemma follows. ◀

Next, let  $u_i$  be bi-valued with the two values being  $0 \leq a < b$ . Obtain  $u'_i$  from  $u_i$  by replacing  $a$  by 0 and  $b$  by 1. Then,  $u'_i$  is equivalent to  $u_i$ , with the shift and scaling being  $a$  and  $b - a$ , respectively. Therefore the bi-valued instance can be transformed to a unit instance, with both having the same equilibria. Now using Lemma 12 we get:

► **Theorem 15.** *There is a strongly polynomial time algorithm for the bi-valued utilities case of the Hylland-Zeckhauser scheme.*

## 5 Characterizing Optimal Bundles

In this section we give a characterization of optimal bundles for an agent at given prices  $p$  which are not necessarily equilibrium prices. This characterization will be used in Section 6.

**Notation.** For each agent  $i$ , let  $G_i^* \subseteq G$  denote the set of goods from which  $i$  derives maximum utility, i.e.,  $G_i^* = \arg \max_{j \in G} \{u_{ij}\}$ . With respect to an allocation  $x$ , let  $B_i = \{j \in G \mid x_{ij} > 0\}$ , i.e., the set of goods in  $i$ 's bundle.

We identify the following four types of optimal bundles.

**Type A bundles:**  $\alpha_i = 0$  and  $\text{cost}(i) < 1$ .

By complementary slackness, optimal goods will satisfy  $u_{ij} = \mu_i$  and suboptimal goods will satisfy  $u_{ij} < \mu_i$ . Hence the set of optimal goods is  $G_i^*$  and  $B_i \subseteq G_i^*$ . Note that the prices of goods in  $B_i$  can be arbitrary, as long as  $\text{cost}(i) < 1$ .

**Type B bundles:**  $\alpha_i = 0$  and  $\text{cost}(i) = 1$ .

The only difference from the previous type is that  $\text{cost}(i)$  is exactly 1.

**Type C bundles:**  $\alpha_i > 0$  and all optimal goods for  $i$  have the same utility.

Recall that good  $j$  is optimal for  $i$  if<sup>6</sup>  $\alpha_i p_j = u_{ij} - \mu_i$ . Suppose goods  $j$  and  $k$  are both optimal. Then  $u_{ij} = u_{ik}$  and  $\alpha_i p_j = u_{ij} - \mu_i = u_{ik} - \mu_i = \alpha_i p_k$ , i.e.,  $p_j = p_k$ . Since  $\alpha_i > 0$ ,

<sup>6</sup> Note that under this case, optimal goods are not necessarily maximum utility goods; the latter may be suboptimal because their prices are too high.

by complementary slackness,  $\text{cost}(i) = 1$ . Further, since  $\text{size}(i) = 1$ , we get that each optimal good has price 1.

**Type D bundles:**  $\alpha_i > 0$  and not all optimal goods for  $i$  have the same utility.

Suppose goods  $j$  and  $k$  are both optimal and  $u_{ij} \neq u_{ik}$ . Then  $\alpha_i p_j = u_{ij} - \mu_i \neq u_{ik} - \mu_i = \alpha_i p_k$ , i.e.,  $p_j \neq p_k$ . Therefore optimal goods have at least two different prices. Since  $\alpha_i > 0$ , by complementary slackness,  $\text{cost}(i) = 1$ . Further, since  $\text{size}(i) = 1$ , there must be an optimal good with price more than 1 and an optimal good with price less than 1. Finally, if good  $z$  is suboptimal for  $i$ , then  $\alpha_i p_z < u_{iz} - \mu_i$ .

## 6 An Example Having Only Irrational Equilibria

Our example has 4 agents  $A_1, \dots, A_4$  and 4 goods  $g_1, \dots, g_4$ <sup>7</sup>. The agents' utilities for the goods are given in Table 1, with rows corresponding to agents and columns to goods.

■ **Table 1** Agents' utilities.

	$g_1$	$g_2$	$g_3$	$g_4$
$A_1$	2	4	0	8
$A_2$	2	3	0	8
$A_3$	2	0	5	0
$A_4$	0	4	5	0

Thus, agents  $A_1$  and  $A_2$  like, to varying degrees, three goods only,  $g_1, g_2, g_4$ , while agents  $A_3$  and  $A_4$  like two goods each,  $\{g_1, g_3\}$  and  $\{g_2, g_3\}$ , respectively. The precise values of the utilities are not that important; the important aspects are: which goods each agent likes, the order between them, and the ratios  $\frac{u_{14}-u_{12}}{u_{12}-u_{11}}$  and  $\frac{u_{24}-u_{22}}{u_{22}-u_{21}}$ . Notice that the latter are unequal.

We show that this example has a unique equilibrium solution with minimum price 0. In this solution, good  $g_1$  has price 0, and all the other goods have positive irrational values. Agents  $A_1, A_3$  and  $A_4$  buy the goods that they like, and  $A_2$  buys  $g_1$  and  $g_4$  only.

Specifically, we show the following:

► **Theorem 16.** *The stated instance has a unique equilibrium. In this equilibrium, the allocations to agents and prices of goods, other than the zero-priced good, are all irrational numbers. The prices are as follows:*

$$p_1 = 0, \quad p_2 = (23 - \sqrt{17})/32, \quad p_3 = (9 + \sqrt{17})/8, \quad p_4 = (69 - 3\sqrt{17})/32.$$

Letting  $r_i = |1 - p_i|$ , the allocations  $x_{ij}$  of each good  $g_j$  to each agent  $A_i$  are as follows:

$$\begin{aligned} A_1 : \quad x_{11} &= 1 - \frac{r_3}{1+r_3} - \frac{r_4}{1+r_4}, & x_{12} &= \frac{r_2}{r_2+r_3}, & x_{13} &= 0, & x_{14} &= \frac{r_4}{1+r_4} \\ A_2 : \quad x_{21} &= \frac{r_4}{1+r_4}, & x_{22} &= 0, & x_{23} &= 0, & x_{24} &= \frac{1}{1+r_4} \\ A_3 : \quad x_{31} &= \frac{r_3}{1+r_3}, & x_{32} &= 0, & x_{33} &= \frac{1}{1+r_3}, & x_{34} &= 0 \\ A_4 : \quad x_{41} &= 0, & x_{42} &= \frac{r_3}{r_2+r_3}, & x_{43} &= \frac{r_2}{r_2+r_3}, & x_{44} &= 0 \end{aligned}$$

<sup>7</sup> It can be shown, by analyzing relations in the bipartite graph on agents and goods with edges corresponding to non-zero allocations, that any instance with 3 agents and 3 goods and rational utilities has a rational equilibrium.

The proof of the theorem is given in the full paper. Even for a small instance like this one, the analysis of the HZ equilibria is not simple. We outline here the main steps. Consider any equilibrium with minimum price 0. We first analyze qualitatively the prices of the goods with respect to 0 and 1: We show that the zero-priced good must be good 1; good 2 must have price strictly between 0 and 1, and goods 3 and 4 must have price strictly greater than 1. Next we characterize qualitatively which goods are bought in non-zero quantity by each agent: We show that every agent buys only goods that she likes. Agents  $A_3$  and  $A_4$  buy both goods that they like, but only one of agents  $A_1, A_2$  buys all three goods that she likes. If  $A_1$  buys  $g_1, g_2, g_4$ , then  $A_2$  buys only  $g_1, g_4$ . If  $A_2$  buys  $g_1, g_2, g_4$ , then  $A_1$  buys only  $g_2, g_4$ . Finally, we analyze quantitatively each of these two cases. In each case, we show that the prices satisfy a system of (nonlinear) equations, and the system has a unique nonnegative solution. Furthermore, the prices determine uniquely the allocation. In the first case where  $A_1$  buys all three goods that she likes, the unique solution and the corresponding allocations are as given in the Theorem. In the second case where agent  $A_2$  buys all three goods, we show that there is no equilibrium: the unique prices that satisfy the equations yield allocations that violate the market clearance conditions (a good is oversold). The analysis uses heavily the primal-dual complementary slackness equations. We refer to the full paper for the details.

► **Remark 17.** Observe that in the equilibrium, the allocations of all four agents are irrational even though each one of them spends their dollar completely and the allocations form a fractional perfect matching, i.e., add up to 1 for each good and each agent.

## 7 Membership of Exact Equilibrium in FIXP

In this section, we will prove that the problem of computing an HZ equilibrium lies in the class FIXP, which was introduced in [16]. This is the class of problems that can be cast, in polynomial time, as the problem of computing a fixed point of an algebraic Brouwer function. Recall that basic complexity classes, such as P, NP, NC and #P, are defined via machine models. For the class FIXP, the role of “machine model” is played by one of the following: a straight line program, an algebraic formula, or a circuit; further it must use the standard arithmetic operations of  $+$ ,  $-$ ,  $*$ ,  $/$ ,  $\min$  and  $\max$ . We will establish membership in FIXP using straight line programs. Such a program should satisfy the following:

1. The program does not have any conditional statements, such as if ... then ... else.
2. It uses the standard arithmetic operations of  $+$ ,  $-$ ,  $*$ ,  $/$ ,  $\min$  and  $\max$ .
3. It never attempts to divide by zero.

A *total problem* is one which always has a solution, e.g., Nash equilibrium and Hylland-Zeckhauser equilibrium. A total problem is in FIXP if there is a polynomial time algorithm which given an instance  $I$  of length  $|I| = n$ , outputs a polynomial sized straight line program which computes a function  $F_I$  on a closed, convex, real-valued domain  $D(n)$  satisfying: each fixed point of  $F_I$  is a solution to instance  $I$ .

Let  $p$  and  $x$  denote the allocation and price variables. We will give a function  $F$  over these variables and a closed, compact, real-valued domain  $D$  for  $F$ . The function will be specified by a polynomial length straight line program using the algebraic operations of  $+$ ,  $-$ ,  $*$ ,  $/$ ,  $\min$  and  $\max$ , hence guaranteeing that it is continuous. We will prove that all fixed points of  $F$  are equilibrium allocations and prices, hence proving that Hylland-Zeckhauser is in FIXP.

**Notation.** We denote the set  $\{1, \dots, n\}$  by  $[n]$ .  $x_i$  denotes agent  $i$ 's bundle. For each agent  $i$ , choose one good from  $G_i^*$  and denote it by  $i^*$ . If  $e$  is an expression, we will use  $(e)_+$  as a shorthand for  $\max\{0, e\}$ .

Domain  $D = D_p \times D_x$ , where  $D_p$  and  $D_x$  are the domains for  $p$  and  $x$ , respectively, with  $D_p = \{p \mid \forall j \in [n], p_j \in [0, n]\}$  and  $D_x = \{x \mid \forall i \in [n], \sum_{j \in G} x_{ij} = 1, \text{ and } \forall i, j \in [n], x_{ij} \geq 0\}$ .

Let  $(p', x') = F(p, x)$ .  $(p, x)$  can be viewed as being composed of  $n + 1$  vectors of variables, namely  $p$  and for each  $i \in [n]$ ,  $x_i$ . Similarly, we will view  $F$  as being composed of  $n + 1$  functions,  $F_p$  and for each  $i \in [n]$ ,  $F_i$ , where  $p' = F_p(p, x)$  and for each  $i \in [n]$ ,  $x'_i = F_i(p, x)$ . The straight line programs for  $F_p$  and  $F_i$  are given in the figures below.

It is easy to see that if  $F_i$  alters a bundle, the new bundle still remains in the domain; in particular,  $\forall i \in [n]$ ,  $\text{size}(i) = 1$ . Similarly, it is easy to see that the output of  $F_p$  is in the domain  $D_p$ .

**Requirements on equilibria.** Observe that  $(p, x)$  will be an equilibrium for the market if, in addition to the conditions imposed by the domain, it satisfies the following:

1.  $\forall j \in [n]$ ,  $\sum_{i \in A} x_{ij} = 1$ .
2.  $\forall i \in [n]$ ,  $\text{cost}(i) \leq 1$ .
3.  $\forall i \in [n]$ ,  $x_i$  is an optimal bundle for  $i$ . Furthermore,  $\text{cost}(i)$  is minimum over all optimal bundles.

Function  $F$  has been constructed in such a way that if any of these conditions is not satisfied by  $(p, x)$ , then  $F(p, x) \neq (p, x)$ , i.e.,  $(p, x)$  is not a fixed point of  $F$ . Equivalently, we show that every fixed point of  $F$  must satisfy all these conditions and is therefore an equilibrium. The converse also holds. That is, we show:

► **Lemma 18.** *Every fixed point  $(x, p)$  of  $F$  is an equilibrium of the matching market. Conversely, every equilibrium  $(p, x)$ , where some good has price 0, is a fixed point of  $F$ .*

■ **Algorithm 2** Straight line program for function  $F_p$ .

- 
1. For all  $j \in [n]$  do:  $p_j \leftarrow \min\{n, \max\{0, p_j + \sum_{i \in A} x_{ij} - 1\}\}$
  2.  $r \leftarrow \min_{j \in [n]} \{p_j\}$
  3. For all  $j \in [n]$  do:  $p_j \leftarrow p_j - r$
- 

The proof of correctness for the function  $F$  is nontrivial, and is given in the full paper. The proof makes essential use of the characterization of the optimal bundles from Section 5. We first show that  $F$  has the following key property: if  $(p, x)$  is a fixed point, then no step of  $F$  will change  $(p, x)$ , i.e., it couldn't be that some step(s) of  $F$  change  $(p, x)$  and some other step(s) change it back, restoring it to  $(p, x)$ . This is easy to check for  $F_p$ . The proof for  $F_i$  is more delicate and uses a potential function argument based on the changes in  $\text{value}(i) = \sum_j u_{ij}x_{ij}$  and  $\text{cost}(i) = \sum_j p_j x_{ij}$  caused by any change in the allocation  $x_i$  in every step of the algorithm for  $F_i$ . Given the key property, we then show that if a pair  $(x, p)$  does not satisfy one of the equilibrium requirements, then some step of  $F$  will change  $(x, p)$ , hence  $(x, p)$  is not a fixed point. We refer to the full paper for the details of the proofs.

Thus, we have:

► **Theorem 19.** *The problem of computing an exact equilibrium for the Hylland-Zeckhauser scheme is in FIXP.*

■ **Algorithm 3** Straight line program for function  $F_i$ .

- 
1.  $r \leftarrow (\sum_j p_j x_{ij} - 1)_+$ .
  2. For all  $j \in [n]$  do:  $x_{ij} \leftarrow \frac{x_{ij} + r \cdot (1 - p_j)_+}{1 + r \cdot \sum_k (1 - p_k)_+}$
  3.  $t \leftarrow (1 - \sum_j p_j x_{ij})_+$
  4. For all  $k \notin G_i^*$  do:
    - a.  $d \leftarrow \min\{x_{ik}, \frac{t}{n^2}\}$
    - b.  $x_{ik} \leftarrow x_{ik} - d$
    - c.  $x_{ii^*} := x_{ii^*} + d$
  5. For all pairs  $j, k$  of goods s.t.  $u_{ij} \leq u_{ik}$  do:
    - a.  $d \leftarrow \min\{x_{ij}, (p_j - p_k)_+\}$
    - b.  $x_{ij} \leftarrow x_{ij} - d/n$
    - c.  $x_{ik} \leftarrow x_{ik} + d/n$
  6. For all triples  $j, k, l$  of goods such that  $u_{ij} < u_{ik} < u_{il}$  do:
    - a.  $d \leftarrow \min\{x_{ik}, ((u_{il} - u_{ik})(p_k - p_j) - (u_{ik} - u_{ij})(p_l - p_k))_+\}$
    - b.  $x_{ik} \leftarrow x_{ik} - d$
    - c.  $x_{ij} \leftarrow x_{ij} + \frac{u_{il} - u_{ik}}{u_{il} - u_{ij}} d$
    - d.  $x_{il} \leftarrow x_{il} + \frac{u_{ik} - u_{ij}}{u_{il} - u_{ij}} d$
  7. For all triples  $j, k, l$  of goods such that  $u_{ij} < u_{ik} < u_{il}$  do:
    - a.  $d := \min(x_{ij}, x_{il}, ((u_{ik} - u_{ij})(p_l - p_k) - (u_{il} - u_{ik})(p_k - p_j))_+)$
    - b.  $x_{ik} := x_{ik} + d$
    - c.  $x_{ij} := x_{ij} - \frac{u_{il} - u_{ik}}{u_{il} - u_{ij}} d$
    - d.  $x_{il} := x_{il} - \frac{u_{ik} - u_{ij}}{u_{il} - u_{ij}} d$
- 

## 8 Membership of Approximate Equilibrium in PPAD

In this section we define approximate equilibria, and show that the problem of computing an approximate equilibrium is in PPAD.

First let us scale the utilities of all the agents so that they lie in  $[0, 1]$ . This can be done clearly without loss of generality without changing the equilibria.

► **Definition 20.** A pair  $(p, x)$  of (non-negative) prices and allocations is an  $\epsilon$ -approximate equilibrium for a given one-sided market if:

1. The total probability share of each good  $j$  is 1 unit, i.e.,  $\sum_i x_{ij} = 1$ .
2. The size of each agent  $i$ 's allocation is 1, i.e.,  $\text{size}(i) = 1$ .
3. The cost of the allocation of each agent is at most  $1 + \epsilon$ .
4. The value of the allocation of each agent  $i$  is at least  $v^*(i) - \epsilon$  where  $v^*(i)$  is the value of the optimal bundle for agent  $i$  under prices  $p$ , i.e. the optimal value of the program: maximize  $\text{value}(i)$ , subject to  $\text{size}(i) = 1$  and  $\text{cost}(i) \leq 1$ . Furthermore, we require that the cost of the allocation  $x_i$  is at most  $c^*(i) - \epsilon$ , where  $c^*(i)$  is the minimum cost of a bundle for agent  $i$  that has the maximum value  $v^*(i)$ .

The corresponding computational problem is: Given a one-sided matching market  $M$  and a rational  $\epsilon > 0$  (in binary as usual), compute an  $\epsilon$ -approximate equilibrium for  $M$ . Polynomial time in this context means time that is polynomial in the encoding size of the

market  $M$  and  $\log(1/\epsilon)$ . We define also a more relaxed version, called a *relaxed  $\epsilon$ -approximate equilibrium* where condition 1 is relaxed to  $|\sum_i x_{ij} - 1| \leq \epsilon$  for all goods  $j$ . It is easy to see that the two versions are polynomially equivalent, i.e., if one can be solved in polynomial time then so can the other.

► **Proposition 21.** *The problems of computing an  $\epsilon$ -approximate equilibrium and a relaxed approximate equilibrium are polynomially equivalent.*

Note however, that in general an  $\epsilon$ -approximate equilibrium may not be close to an actual equilibrium of the matching market. This phenomenon is similar to the case of market equilibria for the standard exchange markets and to the case of Nash equilibria for games.

We will show membership of the approximate equilibrium problem in PPAD by showing that a relaxed approximate equilibrium can be obtained from an approximate fixed point of a variant of the function  $F$  defined in Section 7.

► **Definition 22.** *A weak  $\epsilon$ -approximate fixed point of a function  $F$  (or weak  $\epsilon$ -fixed point for short) is a point  $x$  such that  $\|F(x) - x\|_\infty \leq \epsilon$ .*

Let  $\mathcal{F}$  be a family of functions, where each function  $F_I$  in  $\mathcal{F}$  corresponds to an instance  $I$  of a problem (in our case a one-sided matching market) that is encoded as usual by a string. The function  $F_I$  maps a domain  $D_I$ , to itself. We assume that  $D_I$  is a polytope defined by a set of linear inequalities with rational coefficients which can be computed from  $I$  in polynomial time; this clearly holds for our problem. We use  $|I|$  to denote the length of the encoding of an instance  $I$  (i.e., the length of the string). If  $x$  is a rational vector, we use  $\text{size}(x)$  to denote the number of bits in a binary representation of  $x$ .

► **Definition 23.** *A family  $\mathcal{F}$  of functions is polynomially computable if there is a polynomial  $q$  and an algorithm that, given the string encoding  $I$  of a function  $F_I \in \mathcal{F}$  and a rational point  $x \in D_I$ , computes  $F_I(x)$  in time  $q(|I| + \text{size}(x))$ .*

*A family  $\mathcal{F}$  of functions is polynomially continuous if there is a polynomial  $q$  such that for every  $F_I \in \mathcal{F}$  and every rational  $\epsilon > 0$  there is a rational  $\delta$  such that  $\log(1/\delta) \leq q(|I| + \log(1/\epsilon))$  and such that  $\|x - y\|_\infty \leq \delta$  implies  $\|F_I(x) - F_I(y)\|_\infty \leq \epsilon$  for all  $x, y \in D_I$ .*

It was shown in [16] that, if a family of functions is polynomially computable and polynomially continuous, then the corresponding weak approximate fixed point problem (given  $I$  and rational  $\delta > 0$ , compute a weak  $\delta$ -approximate fixed point of  $F_I$ ) is in PPAD. The family  $\mathcal{F}$  of functions for the online matching market problem defined in Section 7 is obviously polynomially computable. It is easy to check also that it is polynomially continuous.

We will use a variant  $F'$  of the function  $F$  of Section 7, where the functions  $F_i$  for the allocations are modified as follows. Step 5 for all pairs  $j, k$  of goods, and steps 6, and 7 for all triples  $j, k, l$  are applied all independently in parallel to the allocation that results after step 4. In order for the allocation to remain feasible (i.e. have  $x_{ij} \geq 0$  for all  $i, j$ ), we change line 5a in  $F'_i$  to  $d \leftarrow \min\{\frac{x_{ij}}{3}, (p_j - p_k)_+\}$ , change line 6a to  $d \leftarrow \min\{\frac{x_{ik}}{3n^2}, ((u_{il} - u_{ik})(p_k - p_j) - (u_{ik} - u_{ij})(p_l - p_k))_+\}$ , and we change line 7a to  $d \leftarrow \min\{\frac{x_{ij}}{3n^2}, \frac{x_{il}}{3n^2}, ((u_{ik} - u_{ij})(p_l - p_k) - (u_{il} - u_{ik})(p_k - p_j))_+\}$ . In this way, a coordinate  $x_{ij}$  can be decreased by the operations of step 5 for all pairs  $j, k$  at most by  $x_{ij}/3$  in total, and the same is true for the total decrease from the operations of steps 6 and 7 for all triples involving good  $j$ ; therefore, the coordinates  $x_{ij}$  remain nonnegative. The function for the prices remains the same as before. All the properties shown in Section 7 for  $F$  hold also for  $F'$ . The family  $\mathcal{F}'$  of these functions  $F'_I$  is clearly also polynomially computable and polynomially continuous.



Let  $I$  be a given instance of the matching market problem. Every utility  $u_{ij}$  is a rational number, without loss of generality in  $[0, 1]$ , which is given as the ratio of two integers represented in binary. Let  $m$  be the maximum number of bits needed to represent a utility. Note that every nonzero  $u_{ij}$  is at least  $1/2^m$  and the difference between any two unequal utilities is at least  $1/2^{2m}$ . Given a positive rational  $\epsilon$  (wlog in  $[0, 1]$ ), let  $\delta = \epsilon/(n^{10}2^{6m})$ . Note that  $\log(1/\delta)$  is bounded by a polynomial in  $|I|$  and  $\log(1/\epsilon)$ . We show the following:

► **Lemma 24.** *Every weak  $\delta$ -approximate fixed point of  $F_I'$  is a relaxed  $\epsilon$ -approximate equilibrium of the market  $I$ .*

The proof follows and adapts the proof in Section 7 of the analogous statement for the exact fixed points. The proof is rather involved; we refer to the full paper for the details. As a consequence of Lemma 24 and Proposition 21 we have:

► **Theorem 25.** *The problem of computing an  $\epsilon$ -approximate equilibrium of a given matching market is in PPAD.*

## 9 Discussion

As stated in the Introduction, a conclusive proof of intractability of the HZ scheme, via either a proof of FIXP-hardness for exact equilibrium or PPAD-hardness for approximate equilibrium, has eluded us. One of the difficulties is the following: Optimal bundles of agents in an HZ equilibrium may include zero-utility zero-priced goods as “fillers” to satisfy the size constraint, e.g., observe their use in our Algorithm for the case of dichotomous utilities. In this easy setting, we knew which were the “filler” goods. However, when faced with a complex instance of HZ, we don’t a priori know which zero-utility goods will be used as “fillers”. Therefore, even though an agent may have very few positive utility goods, other goods are also in play, thereby giving no “control” on the equilibrium outcome.

We propose exploring the following avenue, in addition to the usual ones, for arriving at evidence of intractability: Relax the notion of polynomial time reducibility suitably and obtain a weaker result than FIXP-hardness or PPAD-hardness.

Other open problems related to our work are: obtain efficient algorithms for computing approximate equilibria, suitably defined, and identify other special cases, besides the bi-valued case, for which equilibrium is easy to compute. Additionally, generalizations and variants of the HZ scheme deserve attention, most importantly to two-sided matching markets [14].

Encouraged by success on the bi-valued utilities case, we considered its generalization to the tri-valued utilities case, in particular,  $\{0, \frac{1}{2}, 1\}$  utilities. We believe even this case has instances with only irrational equilibria. Finding such an example or proving rationality is non-trivial and we leave it as an open problem. Furthermore, it will not be surprising if even this case is intractable; resolving this is a challenging open problem.

---

## References

- 1 Saeed Alaei, Pooya Jalaly Khalilabadi, and Eva Tardos. Computing equilibrium in matching markets. In *Proceedings of the 2017 ACM Conference on Economics and Computation*, pages 245–261, 2017.
- 2 Saugata Basu, Richard Pollack, and MF Roy. A new algorithm to find a point in every cell defined by a family of polynomials. *Quantifier Elimination and Cylindrical Algebraic Decomposition*, B. Caviness and J. Johnson eds., Springer-Verlag, to appear, 1995.
- 3 Garrett Birkhoff. Tres observaciones sobre el algebra lineal. *Univ. Nac. Tucuman, Ser. A*, 5:147–154, 1946.

- 4 Anna Bogomolnaia and Hervé Moulin. A new solution to the random assignment problem. *Journal of Economic theory*, 100(2):295–328, 2001.
- 5 Anna Bogomolnaia and Hervé Moulin. Random matching under dichotomous preferences. *Econometrica*, 72(1):257–279, 2004.
- 6 Eric Budish. The combinatorial assignment problem: Approximate competitive equilibrium from equal incomes. *Journal of Political Economy*, 119(6):1061–1103, 2011.
- 7 Xi Chen, Decheng Dai, Ye Du, and Shang-Hua Teng. Settling the complexity of Arrow-Debreu equilibria in markets with additively separable utilities. In *2009 50th Annual IEEE Symposium on Foundations of Computer Science*, pages 273–282. IEEE, 2009.
- 8 Xi Chen, Xiaotie Deng, and Shang-Hua Teng. Settling the complexity of computing two-player Nash equilibria. *Journal of the ACM (JACM)*, 56(3):1–57, 2009.
- 9 Xi Chen, Dimitris Paparas, and Mihalis Yannakakis. The complexity of non-monotone markets. *J. ACM*, 64(3):20:1–20:56, 2017. doi:10.1145/3064810.
- 10 Xi Chen and Shang-Hua Teng. Spending is not easier than trading: on the computational equivalence of Fisher and Arrow-Debreu equilibria. In *International Symposium on Algorithms and Computation*, pages 647–656. Springer, 2009.
- 11 Constantinos Daskalakis, Paul W Goldberg, and Christos H Papadimitriou. The complexity of computing a Nash equilibrium. *SIAM Journal on Computing*, 39(1):195–259, 2009.
- 12 Nikhil R Devanur and Ravi Kannan. Market equilibria in polynomial time for fixed number of goods or agents. In *2008 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 45–53. IEEE, 2008.
- 13 Nikhil R Devanur, Christos H Papadimitriou, Amin Saberi, and Vijay V Vazirani. Market equilibrium via a primal–dual algorithm for a convex program. *Journal of the ACM (JACM)*, 55(5):22, 2008.
- 14 Federico Echenique, Antonio Miralles, and Jun Zhang. Constrained pseudo-market equilibrium. *arXiv preprint*, 2019. arXiv:1909.05986.
- 15 Federico Echenique, Antonio Miralles, and Jun Zhang. Fairness and efficiency for probabilistic allocations with endowments. *arXiv preprint*, 2019. arXiv:1908.04336.
- 16 K. Etessami and M. Yannakakis. On the complexity of Nash equilibria and other fixed points. *SIAM J. Comput.*, 39(6):2531–2597, 2010.
- 17 Simons Institute for the Theory of Computing. Online and matching-based market design, 2019. URL: <https://simons.berkeley.edu/programs/market2019>.
- 18 Jugal Garg, Ruta Mehta, and Vijay V. Vazirani. Dichotomies in equilibrium computation and membership of PLC markets in FIXP. *Theory of Computing*, 12(1):1–25, 2016.
- 19 Jugal Garg, Ruta Mehta, Vijay V Vazirani, and Sadra Yazdanbod. Settling the complexity of Leontief and PLC exchange markets under exact and approximate equilibria. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 890–901, 2017.
- 20 Jugal Garg, Thorben Tröbst, and Vijay V Vazirani. An Arrow-Debreu extension of the Hylland-Zeckhauser scheme: Equilibrium existence and algorithms. *arXiv preprint*, 2020. arXiv:2009.10320.
- 21 Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric algorithms and combinatorial optimization*, volume 2. Springer Science & Business Media, 2012.
- 22 Yinghua He, Antonio Miralles, Marek Pycia, and Jianye Yan. A pseudo-market approach to allocation with priorities. *American Economic Journal: Microeconomics*, 10(3):272–314, 2018.
- 23 Aanund Hylland and Richard Zeckhauser. The efficient allocation of individuals to positions. *Journal of Political economy*, 87(2):293–314, 1979.
- 24 Kamal Jain. A polynomial time algorithm for computing an Arrow–Debreu market equilibrium for linear utilities. *SIAM Journal on Computing*, 37(1):303–318, 2007. doi:10.1137/S0097539705447384.
- 25 Phuong Le. Competitive equilibrium in the random assignment problem. *International Journal of Economic Theory*, 13(4):369–385, 2017.

- 26 Andy McLennan. Efficient disposal equilibria of pseudomarkets. In *Workshop on Game Theory*, volume 4, page 8, 2018.
- 27 Hervé Moulin. Fair division in the age of internet. *Annual Review of Economics*, 2018.
- 28 C. Papadimitriou. On the complexity of the parity argument and other inefficient proofs of existence. *JCSS*, 48(3):498–532, 1994.
- 29 Alvin E Roth, Tayfun Sönmez, and M Utku Ünver. Pairwise kidney exchange. *Journal of Economic theory*, 125(2):151–188, 2005.
- 30 Lloyd Shapley and Herbert Scarf. On cores and indivisibility. *Journal of mathematical economics*, 1(1):23–37, 1974.
- 31 V. V. Vazirani. The notion of a rational convex program, and an algorithm for the Arrow-Debreu Nash bargaining game. *JACM*, 59(2), 2012.
- 32 V. V. Vazirani and M. Yannakakis. Market equilibria under separable, piecewise-linear, concave utilities. *JACM*, 58(3), 2011.
- 33 Vijay V. Vazirani. Efficient, strategyproof mechanisms for one-sided matching markets. Manuscript, 2020.
- 34 John Von Neumann. A certain zero-sum two-person game equivalent to the optimal assignment problem. *Contributions to the Theory of Games*, 2(0):5–12, 1953.
- 35 Mihalis Yannakakis. Unpublished, 2013.



# An $O(N)$ Time Algorithm for Finding Hamilton Cycles with High Probability

**Rajko Nenadov**

ETH Zürich, Switzerland

raikon@gmail.com

**Angelika Steger**

ETH Zürich, Switzerland

asteger@inf.ethz.ch

**Pascal Su**

ETH Zürich, Switzerland

sup@inf.ethz.ch

---

## Abstract

We design a randomized algorithm that finds a Hamilton cycle in  $\mathcal{O}(n)$  time with high probability in a random graph  $G_{n,p}$  with edge probability  $p \geq C \log n/n$ . This closes a gap left open in a seminal paper by Angluin and Valiant from 1979.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Graph algorithms analysis; Mathematics of computing  $\rightarrow$  Random graphs; Mathematics of computing  $\rightarrow$  Graph algorithms; Mathematics of computing  $\rightarrow$  Matchings and factors; Theory of computation  $\rightarrow$  Random walks and Markov chains

**Keywords and phrases** Random Graphs, Hamilton Cycle, Perfect Matching, Linear Time, Sublinear Algorithm, Random Walk, Coupon Collector

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.60

**Funding** *Pascal Su*: This author was supported by grant no. 200021 169242 of the Swiss National Science Foundation.

## 1 Introduction

A Hamilton cycle is a cycle in a graph that visits every vertex exactly once. Determining whether a graph has a Hamilton cycle is a notoriously difficult problem that has been tackled in various ways. In general, it is known to be  $\mathcal{NP}$ -hard, putting it in a bag of complexity theory together with colorability or SAT, problems for which one has tried to find polynomial time algorithms for a long time without any success so far.

While the Hamilton cycle problem is a difficult problem in general, it turns out that for most graphs it is actually not. To illustrate this, we take a closer look at the Erdős-Rényi random graph  $G_{n,p}$  which is an  $n$ -vertex graph with each edge being present independently with probability  $p$ . The existence question of the Hamilton cycle problem is very well understood, cf. the comprehensive survey by Frieze [13]. Let  $\mathcal{H}$  be the set of Hamiltonian graphs, then for  $G_{n,p}$  it holds that (Komlós and Szemerédi [19] and Korshunov [20])

$$Pr[G_{n,p(n)} \in \mathcal{H}] = \begin{cases} 0, & p(n) = \frac{\log(n) + \log \log(n) - \omega(1)}{n} \\ e^{-e^{-c}}, & p(n) = \frac{\log(n) + \log \log(n) + c + o(1)}{n} \\ 1, & p(n) = \frac{\log(n) + \log \log(n) + \omega(1)}{n}, \end{cases}$$

which is limitwise the same as the threshold for when  $G_{n,p}$  has minimum degree 2. So really vertices of degree one are the bottleneck for random graphs. In fact, it is known that if we add the edges randomly one by one, the moment we reach minimum degree 2 is the same as



© Rajko Nenadov, Angelika Steger, and Pascal Su;

licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 60; pp. 60:1–60:17

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

the moment the graph becomes Hamiltonian with high probability [1]. And this threshold is also robust (e.g. [22, 23]). For other random graph models like the random graph with  $m$  edges  $G_{n,m}$ , the random regular graph  $G_{n,r}$  or the  $k$ -out which takes  $k$  random edges from every vertex the corresponding thresholds for Hamiltonicity are also known [7, 9, 11, 26, 29]. Similar to the classical random graph case also in these cases the thresholds coincides with a local obstruction such as minimum degree two or any two vertices have a neighborhood of size at least 3. And this is not a coincidence. Randomness gives us such nice expansion properties that only the small structures can be an obstruction to the Hamilton cycle. This phenomenon has been observed also for other properties such as connectivity, containing a perfect matching or colorability.

The proofs of Komlos and Szemerédi and Korshunov are just existential, i.e. they determine the threshold for the existence of Hamilton cycle, but do not provide an efficient algorithm for finding it. In a seminal paper, Angluin and Valiant [4] show that with the input given as a random adjacency list one can find Hamilton cycles in  $G_{n,p}$  for  $p \geq C \log n/n$  in  $\mathcal{O}(n \log^2 n)$  time with high probability. There are two ways in which this result is possibly non-optimal: the lower bound on  $p$  and the runtime. The first point was considered by Shamir and then Bollobas, Fenner and Frieze, who brought the bound down to the existence threshold of  $G_{n,p}$ . In more recent works the runtime has also been optimized for graphs given in adjacency matrix form, assuming a pair of vertices can be queried in constant time. We summarize these results in the table below. There are various related results that are hard to compare, as their setting is slightly different [2, 12, 14, 15]. Some of the results are assuming the graph is given as an adjacency matrix with black box queries and the runtime  $\mathcal{O}(n/p)$  is optimal in that model.

Authors	Year	Time	$p(n)$	Graph Model
Angluin, Valiant [4]	'79	$\mathcal{O}(n \log^2(n))$	$p \geq \frac{C \log(n)}{n}$	adj. list
Shamir [28]	'83	$\mathcal{O}(n^2)$	$p \geq \frac{\log(n) + (3+\epsilon) \log \log(n)}{n}$	adj. list
Bollobas, Fenner, Frieze [8]	'87	$n^{4+o(1)}$	$p \geq \textit{Existence threshold}$	adj. list
Gurevich, Shelah [16]	'87	$\mathcal{O}(n/p)$	$p \text{ const.}$	adj. matrix
Thomason [30]	'89	$\mathcal{O}(n/p)$	$p \geq Cn^{-1/3}$	adj. matrix
Alon, Krivelevich [3]	'20	$\mathcal{O}(n/p)$	$p \geq 70n^{-1/2}$	adj. matrix

In this paper we consider the second question that was left open in the Angluin-Valiant paper: can the runtime be improved. Note that a graph with  $p \geq C \log n/n$  has  $\Theta(n \log n)$  edges. Thus, improving the runtime below this bound requires a *sublinear* algorithm, i.e. sublinear in the input size. These are algorithms that produce an output without reading the input completely (see e.g. [27] for an overview of the topic). Such algorithms are less restrictive than those designed for online or a (semi-)streaming model as they allow some control over which part of an input is used. However for graphs with  $n$  vertices and  $m \gg n$  edges the algorithm is only allowed to read  $o(m)$  edges, i.e., a negligible fraction of the input – but nevertheless has to compute the desired output correctly.

## 1.1 Our contribution

In this paper we show that given a random graph with edge probability  $p \geq C \log n/n$ , for an appropriately chosen constant  $C$ , we can find a Hamilton cycle in  $\mathcal{O}(n)$  time with high probability. This time is clearly optimal, as the algorithm has to return  $\Omega(n)$  edges. We assume that the graph is given to us with randomly ordered adjacency lists, such that we can query the next neighbor in those lists for any vertex in constant time.



■ **Figure 1** Algorithm uses a random walk like strategy, blue edge is the `newneighbor()`.

► **Theorem 1.** *There exists a randomized algorithm  $\mathcal{R}$  which finds a Hamilton cycle in a random graph  $G_{n,p}$  in  $\mathcal{O}(n)$  time with high probability, provided  $p \geq C \log n/n$  for a sufficiently large constant  $C$ .*

Note that “with high probability” is always meant to mean with probability  $1 - o(1)$  tending to one as  $n$  tends to infinity and takes into account all sources of randomness: i.e., the randomness of the algorithm, the random graph and the randomness of the datastructure used to store the graph (random ordering of the adjacency lists).

Our paper is organized as follows. Section 2 contains the algorithm and the proof of Theorem 1. It is based on three technical lemmas that we prove in Section 3.

## 2 Algorithm

The most commonly used technique for efficient cycle extensions is Posa rotations. This is also the case for the original algorithm of Angluin and Valiant [4], which we outline in Section 2.1 below, cf. also Figure 2. To reduce the runtime to  $\mathcal{O}(n)$  we reduce the total number of Posa rotations that are required and simultaneously also restrict ourselves to certain types of Posa rotations so that we can realize each of them in  $\mathcal{O}(\log n)$  time.

### 2.1 Finding A Hamilton Cycle via Posa Rotations

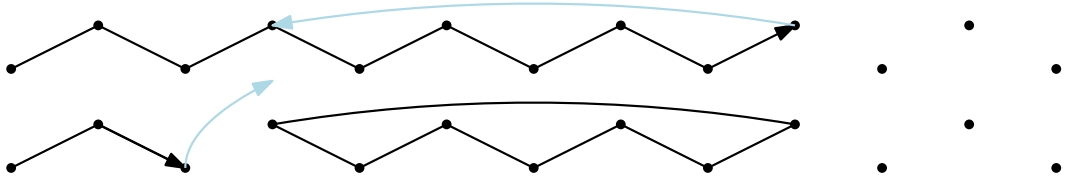
We sketch here the algorithm of Angluin and Valiant. The main idea of their algorithm is to perform a greedy random walk until all vertices are incorporated in the path/cycle. This means we start from an arbitrary vertex and query a neighbor of that vertex. If the neighbor is already contained in the path we have built so far we consider this a failure and we query a new neighbor. Otherwise we add the neighbor to the path and continue from the new endpoint vertex (see Figure 1).

Once the path is long enough (at least  $n/2$ ) we add possible Posa rotations. Assume we start with a path  $P = (v_1, \dots, v_s)$ , then if we find two edges such that for some index  $i \in [s]$  the edges are of the form  $\{v_{i+1}, v_s\}$  and  $\{v_i, v_j\}$  for some  $j > i + 1$ , we can rearrange the path to form a new path  $P' = (v_1, \dots, v_i, v_j, v_{j+1}, \dots, v_s, v_{i+1}, \dots, v_{j-1})$  and now the new path has the same vertex set but a different endpoint vertex. This we call a Posa rotation. Additionally we will always want *long* Posa rotations meaning  $s - i$  must be at least  $n/2$  to ensure that we can find the second edge needed quickly with high probability.

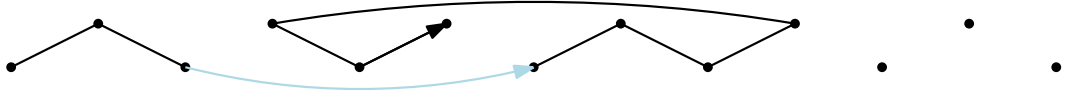
So during our Algorithm if the neighbor  $(v_{i+1})$  of the endpoint of the path  $(v_s)$  has distance at least  $n/2$  from the endpoint along the path we use that edge to build a cycle and continue from the vertex preceding the neighbor  $(v_i)$  on the path (see Figure 2). This leaves a cycle of size at least  $n/2$  and if we ever find one of the vertices on the cycle to be the neighbor of the current endpoint we reincorporate the large cycle by appending it to the path (again giving a new endpoint).

Many details need to be considered on how random variables interact, etc., but leaving those aside one can easily convince oneself that on average the current vertex changes after a constant number of queries to a new random vertex, and that the number of queries until





■ **Figure 2** Posa rotation, detaching a large cycle.



■ **Figure 3** Reincorporating the large cycle.

the path length increases by one is geometrically distributed and has an expectation of  $n/(n-i)$  where  $i$  is the current length of the path. The total number of Posa rotations is thus bounded by

$$\mathcal{O}\left(\sum_{i=1}^n \frac{n}{i}\right) = \mathcal{O}(n \log n).$$

As each Posa rotation takes time  $\log n$  to realize this gives a total running time of  $\mathcal{O}(n \log^2 n)$ .

## 2.2 Our Algorithm

We give a short overview of the new algorithm we propose. The algorithm comes in two phases. In phase 1 we find two random perfect matchings. The union of these two random perfect matchings forms a two regular graph, i.e., a set of disjoint cycles or double edges covering all vertices. It is not difficult to show that the number of cycles is with high probability bounded by  $2 \log n$ . In phase 2 of the algorithm we stitch these  $2 \log n$  cycles together.

For the analysis of the algorithm it is very helpful to assume that a query for a new neighbor of some vertex  $v$  returns a vertex  $w$  that is *uniformly* distributed over all vertices in  $V - v$  and *independent* from all previous queries. Of course such an assumption a priori does not hold if we simply return the next vertex from the adjacency list of  $v$ . We realize this by directing the edges and resampling. More formally, we will show the following lemma in Section 3; in the remainder of Section 2 we will use the corresponding function `newneighbor()` as a black box.

► **Lemma 8** (`newneighbor`). *It is possible to interact with the graph  $G_{n,p}$ ,  $p \geq \frac{C \log n}{n}$ , with an algorithmic procedure `newneighbor( $v$ )` which has the following properties with high probability:*

- (i) *Calling `newneighbor( $v$ )` returns a neighbor of  $v$  distributed uniformly among  $V - v$  and independent of all calls so far – as long as we make at most  $\mathcal{O}(n)$  calls to `newneighbor()` altogether and every vertex is queried at most  $100 \log n$  times.*
- (ii) *The total run time of all  $\mathcal{O}(n)$  calls is  $\mathcal{O}(n)$ .*

Note that this algorithm uses both internal randomness as well as the randomness of  $G_{n,p}$ . If `newneighbor( $v$ )` ever returns 'there are no more neighbors' we immediately terminate the entire algorithm and return failure. To avoid this, we will prove that we query `newneighbor( $v$ )` from any vertex at most  $100 \log n$  times w.h.p. and choose  $C$  large enough so that with high probability the minimum degree of the random graph is large enough.

### 2.2.1 Phase 1: Perfect Matching

In the first phase of the algorithm we show that we can find a perfect matching in  $\mathcal{O}(n)$  time. We call the algorithmic procedure described in this section **FastPerfectMatching**, see Algorithm 1. In fact, for an easier understanding of the required ideas, we work in this section with a random *bipartite* random graph. This can easily be done by partitioning the vertex set  $V$  into two equal sets  $A$  and  $B$  arbitrarily (if  $n$  is odd we set one vertex aside and include it in phase 2) and only considering the edges between  $A$  and  $B$ . Formally, the function **newABneighbor**( $v$ ) calls **newneighbor**( $v$ ) until we receive a neighbor which is in  $B$  (resp.  $A$ ).

▷ **Claim 2.** If we call **newABneighbor**() for a sequence of  $\mathcal{O}(n)$  vertices, in which every vertex  $v \in A \cup B$  occurs at most  $\log n$  times, then with high probability this results in at most  $\mathcal{O}(n)$  calls to **newneighbor**() with at most  $6 \log n$  calls per vertex.

The claim holds because any call of **newneighbor**() has probability at least  $1/2$  to be in the correct partition and, by our assumptions on **newneighbor**(), the calls are independent. We can thus apply concentration bounds for binomial distributions and union bound for every vertex. Clearly, **newABneighbor**() still has a uniform and independent distribution over all vertices of the opposite partition.

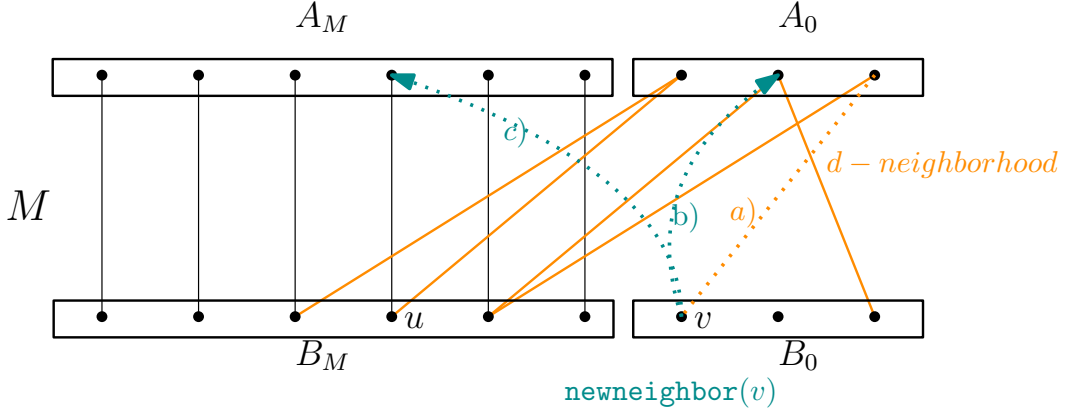
Let  $G$  be the balanced bipartite graph with partitions  $A$  and  $B$ . During the algorithm we will maintain a matching  $M$  which covers some of the vertices and is empty at first. At any point in time, we denote by  $A_M$  the vertices in  $A$  that are covered by the matching and with  $A_0$  the unmatched vertices. Equivalently for  $B_M$  and  $B_0$ .

Additionally we need a set of edges that expand well from the vertices of  $A$ . And we need to be able to keep track of them efficiently and on the fly. So for any vertex  $v$  we define the  $d$ -neighborhood of  $v$ ,  $N_d(v) \subseteq V(G)$ , to be the set of the first  $\lceil d \rceil$  calls to the function **newABneighbor**( $v$ ). In particular this implies that for any  $d' < d$  the  $d'$ -neighborhood is contained in the  $d$ -neighborhood of  $v$ . Similarly, the  $d$ -neighborhood of a set of vertices  $S$ , denoted by  $N_d(S)$ , is defined as the union of the  $d$ -neighborhoods of all vertices in  $S$ . We expose and keep track of the  $d$ -neighborhood of the unmatched vertices  $A_0$ ,  $N_{d(|A_0|)}(A_0)$ , for the function  $d(t) = \min(\sqrt{n/t}, \log n)$ . This gives us a neighborhood large enough for the random walks to be effective, but small enough so that we do not need too much time to update/expose.

To increase the matching we call a subroutine **IncreaseMatching**. **IncreaseMatching** takes as argument the current matching  $M$  and an unmatched vertex  $v \in B_0$ . It proceeds as follows. If  $v$  is in  $N_d(A_0)$  we add the corresponding neighbor in  $A_0$  and  $v$  to the matching. If not we take  $w = \text{newABneighbor}(v)$ . If  $w$  is in  $A_0$  we add the edge  $\{w, v\}$  to  $M$ . If neither of the two is the case, then  $w \in A_M$  and there exists a unique  $u$  such that  $\{w, u\}$  is currently in  $M$ . We swap  $\{w, u\}$  for  $\{w, v\}$ , thereby making  $u$  a new unmatched vertex, and repeat **IncreaseMatching** with  $u$ , cf. Figure 4.

Clearly, during the run of the algorithm we also have to dynamically update the  $d$ -neighborhood of  $A_0$ . In particular this means removing  $N_d(w)$  of a newly matched vertex  $w$  and, if  $d(|A_0|)$  increases, adding vertices from additional calls to **newABneighbor**() for every vertex in  $A_0$ .

To bound the runtime of Algorithm 2, **FastPerfectMatching**, we observe first that we increase the matching exactly  $n$  times, which is inline with our desired bound of  $\mathcal{O}(n)$ . We can thus concentrate on bounding the *recursive* calls to **IncreaseMatching** in line 13 of **IncreaseMatching**.



■ **Figure 4** For `IncreaseMatching` three things can happen. Either a) the vertex is already in the neighborhood of  $A_0$ , in which case we match immediately, b) the vertex `newABneighbor(v)` is in  $A_0$ , which also gets matched, or c) `newABneighbor(v)` is in  $A_M$ . Then we swap the matching and continue from the partner of the `newABneighbor(v)`.

■ **Algorithm 1** `FastPerfectMatching(G)`.

---

```

1:  $B_0 \leftarrow B$ ;  $B_M \leftarrow \{\}$ ;  $A_0 \leftarrow A$ ;  $A_M \leftarrow \{\}$ ;
2:  $d \leftarrow 0$ ;  $M \leftarrow \{\}$ 
3: while  $B_0 \neq \{\}$  do
4:    $v \leftarrow$  arbitrary vertex from  $B_0$  ▷ and remove from  $B_0$ 
5:   IncreasingMatching( $G, M, v$ ); ▷ see Algorithm 2
6:   while  $d < \min\left(\sqrt{\frac{n}{|A_0|}}, \log(n)\right)$  do
7:      $d \leftarrow d + 1$ 
8:     Add newABneighbor(v) to the  $d$ -neighborhood for every vertex in  $v \in A_0$ 
9: return Matching  $M$ 

```

---

► **Lemma 3.** Let  $\mathcal{L}_i$  denote the number of calls `IncreaseMatching` in line 13, while  $|A_0| = i$  for any  $i \in [n]$ . Then the  $\mathcal{L}_i$  are dominated by independent geometric distributions with success probability  $p_i = \frac{i \cdot d(i)}{100n}$ .

**Proof.** Whenever we are at a vertex  $v$  in  $B$  we expose an edge to a random neighbor in the set  $A$ . If that vertex is in  $A_0$  we match  $v$  and  $|A_0|$  decreases by one so we end the count of  $\mathcal{L}_{|A_0|}$ . Otherwise we swap with a matched vertex and get a new starting point in  $B_0$ . As `newABneighbor()` is independent and uniform, and the matching forms a bijection between  $A_M$  and  $B_M$ , the fact that the vertex is not in  $A_0$ , implies that we get a new *random* vertex  $u$  in  $B_M$  for the next call. If this vertex is in the exposed  $d$ -neighborhood of  $A_0$  we stop and match to a vertex in  $A_0$  also ending the count of  $\mathcal{L}_{|A_0|}$ .

To assess the probability of stopping, we use the *expansion properties* of the  $d$ -neighborhood of  $A_0$  that are inherited from the random graph. This means in particular that the exposed neighborhood of  $A_0$ ,  $N_{d(|A_0|)}(A_0)$ , has size at least  $\frac{1}{100}|A_0| \cdot d(|A_0|)$ , cf. Lemma 9 in Section 3 for a proof. The probability of hitting a vertex in  $A_0$  or the  $d$ -neighborhood of  $A_0$  (while looking at the matched vertex of  $w$  in  $B_M$ ) is thus at least  $\frac{|A_0| \cdot d(|A_0|)}{100n}$ . Every new call of `newABneighbor()` is independent by Lemma 8, thus  $\mathcal{L}_i$  is dominated by an independent geometric distribution with success probability as claimed. ◀

---

**Algorithm 2** IncreaseMatching( $G, M, v$ ).
 

---

```

1: if  $v \in N_d(A_0)$  then
2:    $w \leftarrow [\text{neighbor of } v] \in A_0$ 
3:   Add  $\{v, w\}$  to  $M$ 
4:   Remove  $w$  from  $A_0$  and update  $N_d(A_0)$ 
5:   return
6:  $w \leftarrow \text{newABneighbor}(v)$ 
7: if  $w \in A_0$  then
8:   Add  $\{v, w\}$  to  $M$ 
9:   Remove  $w$  from  $A_0$  and update  $N_d(A_0)$ 
10:  return
11:  $u \leftarrow$  unique vertex with  $\{u, w\} \in M$ 
12: Remove  $\{u, w\}$  from  $M$  and replace with  $\{v, w\}$ 
13: IncreaseMatching( $G, M, u$ )
14: return
  
```

---

We are now ready to prove the desired complexity bound:

► **Proposition 4.** *FastPerfectMatching finds a perfect matching in a balanced random bipartite graph in time  $\mathcal{O}(n)$  with high probability.*

**Proof.** There are two main contributions to the running time of the Algorithm. First the subroutine **IncreaseMatching**, which we prove to be fast with the help of Lemma 3, and secondly the updating and revealing of the  $d$ -neighborhood.

Recall that  $\mathcal{L}_i$  is the random variable corresponding to the number of calls of **IncreaseMatching** in line 13, while  $|A_0| = i$  for any  $i \in [n]$ . We set  $\mathcal{L} = \sum_{i=1}^n \mathcal{L}_i$ . Note that we can ignore the calls in line 5 of **FastPerfectMatching**, as these add only at total of  $\mathcal{O}(n)$  to the run time. From Lemma 3 we know that there exists a coupling to a geometrically distributed random variable  $\mathcal{L}'$  such that  $\mathcal{L}'_i \succeq \mathcal{L}_i$  and  $\mathcal{L}'_i$  is geometrically distributed with  $p_i = \frac{i \cdot d(i)}{100n}$ .

From the definition of  $\mathcal{L}'_i$  we know that  $\mathbb{E}[\mathcal{L}'_i] = \frac{100n}{i \cdot d(i)}$  and  $\text{Var}[\mathcal{L}'_i] = \frac{1-p_i}{p_i^2} \leq \frac{1}{p_i^2} \leq (\frac{100n}{i \cdot d(i)})^2$ . Recall that  $d(i) = \sqrt{n/i}$  whenever  $i \geq \frac{n}{(\log n)^2}$ . The total time used for those sets can thus be bounded in expectation by

$$\sum_{i=\frac{n}{(\log n)^2}}^n \mathbb{E}[\mathcal{L}'_i] = \mathcal{O}\left(\sum_{i=1}^n \frac{\sqrt{n}}{\sqrt{i}}\right) = \mathcal{O}(n),$$

as  $\sum_{i=1}^n i^{-1/2} \leq \int_0^n x^{-1/2} dx = 2\sqrt{n}$ . If  $i \leq \frac{n}{(\log n)^2}$ , then  $d(i) = \log n$ , and the total expected time used for these sets is thus bounded by

$$\sum_{i=1}^{\frac{n}{(\log n)^2}} \mathbb{E}[\mathcal{L}'_i] = \mathcal{O}\left(\sum_{i=1}^n \frac{n}{i \cdot \log(n)}\right) = \mathcal{O}(n).$$

We thus have that  $\mathbb{E}[\mathcal{L}'] = \Theta(n)$  as well. To show that the actual run time is concentrated around the expectation we apply Chebyshev's inequality. A similar case distinction as above gives us

$$\text{Var}[\mathcal{L}'] \leq \sum_{i=\frac{n}{(\log n)^2}}^n \frac{10000n}{i} + \sum_{i=1}^{\frac{n}{(\log n)^2}} \frac{10000n^2}{i^2 \cdot (\log n)^2} = \mathcal{O}\left(\frac{n^2}{(\log n)^2}\right).$$

By Chebyshev's inequality we thus get

$$\Pr[\mathcal{L} \geq 2\mathbb{E}[\mathcal{L}']] \leq \Pr[\mathcal{L}' \geq 2\mathbb{E}[\mathcal{L}']] \leq \frac{\text{Var}[\mathcal{L}']}{(\mathbb{E}[\mathcal{L}'])^2} \leq \mathcal{O}\left(\frac{1}{(\log n)^2}\right),$$

which concludes the first part of the proof.

To bound the time needed to expose the  $d$ -neighborhoods, we observe first that we can order the vertices in  $A$  by the order in which they join the matching. As  $N_{d'}(v) \subseteq N_d(v) \forall d' \leq d$ , we thus have to expose for the  $i$ -th vertex in this ordering at most  $d(n-i) + 1$  edges, where  $d(x) = \min\{\sqrt{n/x}, \log n\}$ . Thus, the total number of exposed edges is bounded by

$$\begin{aligned} \sum_{i=1}^n (d(n-i) + 1) &= \sum_{i=1}^{\frac{n}{(\log n)^2}} (d(i) + 1) + \sum_{i=\frac{n}{(\log n)^2}}^n (d(i) + 1) \\ &\leq \sum_{i=1}^{\frac{n}{(\log n)^2}} (\log n + 1) + \sum_{i=\frac{n}{(\log n)^2}}^n \sqrt{\frac{n}{i}} \leq 4n. \end{aligned}$$

Additionally we show the number of calls to the `newABneighbor()` function is at most  $\log n$  for every vertex w.h.p.. For any  $v \in B$  we call `newABneighbor(v)` exactly once for each time it appears as the matched partner of `newABneighbor(v)`. As the distribution on the neighbors is uniform on  $A$  and we only use `IncreaseMatching`  $\mathcal{O}(n)$  many times in total, the probability that  $v \in B$  occurs at least  $\log n$  times is at most

$$\left(\frac{\mathcal{O}(n)}{\log n}\right) \left(\frac{1}{n}\right)^{\log n} = \mathcal{O}(n^{-2}),$$

with room to spare. We can thus apply a union bound over all vertices in  $B$  to see that w.h.p. no vertex in  $B$  has more than  $\log n$  calls to `newABneighbor()`. Clearly the same holds for vertices in  $A$ , as we only expose the  $d$ -neighborhood and  $d(|A_0|) \leq \log n$  always. This concludes the proof of Proposition 4.  $\blacktriangleleft$

### 2.2.2 Phase 2: Incorporating the Cycle Factor

In the previous section we have seen that we can find a perfect matching in  $\mathcal{O}(n)$  time. In this section we show how we can extend this algorithm to find a Hamilton cycle. To do this we first call the perfect matching algorithm *twice*, resetting the  $d$ -neighborhoods after the first run. By our assumption on the independence on the calls to the function `newneighbor()`, we thereby get two *independent* random perfect matchings. Their union forms a union of cycles (or double edges) covering all vertices (if the number of vertices was odd we add the single vertex excluded in phase 1 here back as a cycle with one vertex). Our task in this phase is to join these cycles into a single cycle. We start with a lemma that bounds the number of cycles that we need to join.

► **Lemma 5.** *The union of two random independent perfect matchings in a bipartite graph contains at most  $2 \log n$  cycles with high probability.*

**Proof.** We claim that the two independent perfect matchings can be seen as a random permutation of  $[n/2]$ . Indeed, without loss of generality we may assume that  $M_1$  is just the identity (by renumbering the vertices appropriately).  $M_1$  and  $M_2$  are independent which implies  $M_2$  corresponds to a random assignment of  $B$  to  $A$ . The union of the two matchings thus defines a random permutation of  $A$ .

For random permutations the number of cycles has been well studied and is related to the Stirling numbers of the first kind. Using a double counting argument one can easily see that the expected number of cycles of length  $2k$  will be  $1/k$ . The total expected number of cycles is thus equal to the  $n$ th harmonic number. It is also well-known that this random variable is concentrated, see e.g. [5] or [6, 21]. Thus, with high probability the number of cycles is bounded by  $2 \log n$ , as claimed.  $\blacktriangleleft$

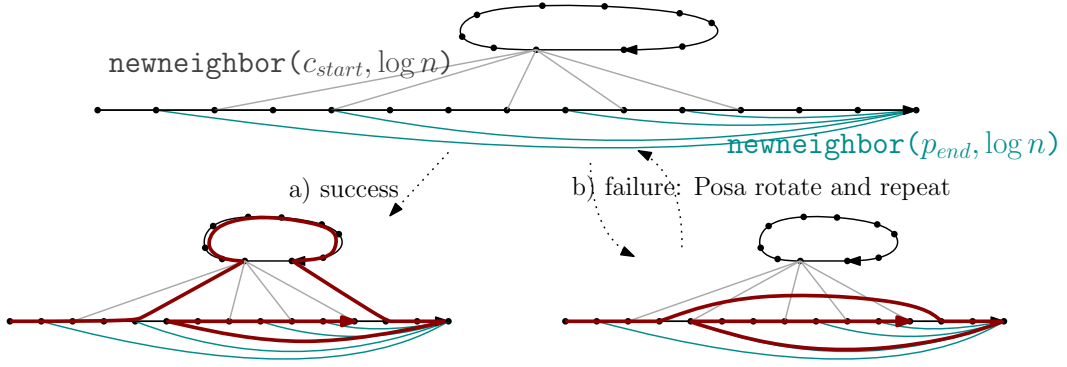
**Description of Algorithm 3 JoinCycles.** To glue the cycles together we proceed in three phases. First we greedily combine cycles into a path, until this path has length at least  $3n/4$ . Then we incorporate the remaining cycles one by one using Algorithm 4 AddSingleCycle. Finally, we close the Hamilton path into a Hamilton cycle (Lemma 7).

The idea behind the first phase is straightforward. We start with an arbitrary cycle and break it apart into a path  $P$ . Consider the endvertex  $p_{end}$  of that path. We use `newneighbor()` to query a new neighbor of  $p_{end}$ . If that neighbor is in a new cycle (which will happen with probability at least  $1/4$ , as long as the path  $P$  contains at most  $3n/4$  vertices), we attach that cycle to  $P$ , thereby also getting a new endpoint  $p_{end}$ . If the latter did not happen, we query a new neighbor. In order to ensure that we do not query too many vertices from a single vertex, we repeat the query for new neighbors at most  $40 \log n$  times. If we have not been successful by then, we give up. It is easy to see that the probability for ever giving up at this stage of the algorithm is bounded by  $o(1)$ . It is also easy to see that the total time spent until the path has length at least  $3n/4$  is bounded by  $\mathcal{O}(n)$ .

Once the path has length at least  $3n/4$ , the probability that a new neighbor is in one of the remaining cycles gets too small (for our purpose) and we thus change strategy. In particular, we add long Posa rotations, so that we can try various endpoints. This is the purpose of the procedure AddSingleCycle (Algorithm 4).

We use a set  $U$  to keep track of *used* vertices. Those are vertices for which we already queried neighbors within the algorithm JoinCycles. We denote the current path by  $P = (p_{start}, \dots, p_{end})$ . We also assume that we have access to a function  $pred_P(v)$  that determines the vertex before  $v$  on the path (null for  $p_{start}$ ), and a function  $half_P(v)$  which is true iff  $v$  is in the first half of  $P$ . We denote the cycle  $C$  that we want to add as  $C = (c_{start}, \dots, c_{end})$ , where  $c_{start}$  is an arbitrary vertex at which we cut  $C$  into a path. We now explore neighborhoods of vertices at once. To do this we denote by `newneighbor`( $v, 40 \log n$ ) the set of vertices that we obtain if we apply `newneighbor`( $v$ )  $40 \log n$  times. Let  $N_{start} = P \cap \text{newneighbor}(c_{start}, 40 \log n)$  and  $N_{end} = P \cap \text{newneighbor}(c_{end}, 40 \log n)$  denote the intersections of these neighborhood vertices with the path  $P$ . Until the cycle  $C$  is part of the path  $P$  we do the following (Figure 5). Let  $N(p_{end}) = \text{newneighbor}(p_{end}, 40 \log n)$  and check for all  $v \in N(p_{end})$  if  $pred_P(v) \in N_{start}$ . If so we also check if  $half_P(v)$  is true.

If we find a vertex  $v$  for which both conditions hold, we join the cycle here. To do this look at  $N_{end}$  and take a vertex  $q \in \text{newneighbor}(c_{end}, 40 \log n)$  such that  $half_P(q)$  is false and  $pred_P(q) \notin U$ . Then we add the cycle to the path by constructing the new path  $P_{new} = (p_{start}, \dots, pred_P(v)) + (c_{start}, \dots, c_{end}) + (q, \dots, p_{end}) + (v, \dots, pred_P(q))$ . Then add  $p_{end}$ ,  $c_{start}$  and  $c_{end}$  to the used vertices  $U$ . (If we cannot find  $q$  we abort the algorithm; it will be easy to show that the probability that this happens is negligible.)



■ **Figure 5** Incorporating a single new cycle with `AddSingleCycle`. The dark red path indicating the new Path after an iteration of the while loop.

■ **Algorithm 3** `JoinCycles( $G, C = M_1 \cup M_2$ )`.

---

```

1:  $U \leftarrow \{\}$ 
2:  $C_0 \leftarrow$  first cycle of  $M_1 \cup M_2$ ,  $(c_{0,\text{start}}, \dots, c_{0,\text{end}})$ ;
3:  $P \leftarrow (c_{0,\text{start}}, \dots, c_{0,\text{end}})$ ;
4:  $p_{\text{end}} \leftarrow$  last vertex of  $P$ ;
5: while  $|P| \leq \frac{3n}{4}$  do
6:    $N \leftarrow \text{newneighbor}(p_{\text{end}}, 40 \log n)$ ;
7:    $U \leftarrow$  add  $p_{\text{end}}$ ;
8:    $v \leftarrow$  Search  $N$  for  $v$  such that  $v \notin P$ 
9:    $(v, \dots, c_{i,\text{end}}) \leftarrow$  cycle of  $v$ ;
10:   $P \leftarrow P + (v, \dots, c_{i,\text{end}})$ ;
11:   $p_{\text{end}} \leftarrow c_{i,\text{end}}$ ;
12: while  $|P| \neq n$  do
13:    $C_i \leftarrow$  any cycle not in  $P$ 
14:   AddSingleCycle( $G, P, C_i, U$ ) ▷ See Algorithm 4
15: return
16: // If any of the “Search” parts of the algorithm fail, we abort the algorithm and return
    failure.
```

---

If the check fails for all  $v \in N(p_{\text{end}})$  we perform a Posa rotation. To do this we take a  $v \in N(p_{\text{end}})$ ,  $v \neq p_{\text{start}}$ , such that  $\text{half}_P(v)$  is true and such that  $\text{pred}_P(v) \notin U$ , and then take a  $q \in \text{newneighbor}(\text{pred}_P(v), 40 \log n)$  such that both  $\text{half}_P(q)$  is false and  $\text{pred}_P(q)$  is unused. We then use  $v$  and  $q$  to construct a new path with a new endpoint, namely  $P_{\text{new}} = (p_{\text{start}}, \dots, \text{pred}_P(v)) + (q, \dots, p_{\text{end}}) + (v, \dots, \text{pred}_P(q))$ . Now we can repeat the above procedure with  $P_{\text{new}}$  and the new endpoint  $p_{\text{newend}} = \text{pred}_P(q)$ . (If we cannot find  $v$  or  $q$  we abort the algorithm; again it will be easy to show that the probability that this happens is negligible.)

To store the path and cycles we use AVL trees with a linked list. The linked list just stores the vertices in the order as they appear in the path resp. cycle. For the AVL tree we take the ordering in the path/linked list as an ordering of the vertices. With this ordering at hand, the AVL tree is well defined, and it allows for searching resp. answering the query  $\text{half}(v)$  in  $\mathcal{O}(\log n)$  time. In addition, splitting the path resp. concatenating two paths correspond to splitting an AVL tree at a given vertex (into a tree containing all smaller vertices and a



---

**Algorithm 4**  $\text{AddSingleCycle}(G, P, C_i, U)$ .

---

```

1: // Function  $\text{half}_P(v)$  returns true if and only if  $v$  is in the first half of  $P$ ;
2: // For any vertex  $v \in P$ ,  $\text{pred}_P(v)$  denotes the vertex before  $v$  on the path  $P$ ;
3:  $p_{\text{end}} \leftarrow$  last vertex of  $P$ ;
4:  $N_{\text{start}} \leftarrow \text{newneighbor}(c_{\text{start}}, 40 \log n) \cap P$ ;
5:  $N_{\text{end}} \leftarrow \text{newneighbor}(c_{\text{end}}, 40 \log n)$ ;
6:  $U \leftarrow$  add  $c_{\text{start}}$  and  $c_{\text{end}}$ ;
7: while true do
8:    $N \leftarrow \text{newneighbor}(p_{\text{end}}, 40 \log n)$ ;
9:    $U \leftarrow$  add  $p_{\text{end}}$ ;
10:  if  $\exists v \in N$  s.t.  $\text{pred}_P(v) \in N_{\text{start}}$  and  $\text{half}_P(v) = \text{true}$  then
11:     $q \leftarrow$  Search  $N_{\text{end}}$  for  $q$  such that  $\text{half}_P(q) = \text{false}$  and  $\text{pred}_P(q) \notin U$ ;
12:     $P \leftarrow (p_{\text{start}}, \dots, \text{pred}_P(v)) + (c_{\text{start}}, \dots, c_{\text{end}}) + (q, \dots, p_{\text{end}}) + (v, \dots, \text{pred}_P(q))$ ;
13:    return
14:  else
15:     $v \leftarrow$  Search  $N$  for  $v$  such that  $\text{half}_P(v) = \text{true}$ ;
16:     $N \leftarrow \text{newneighbor}(\text{pred}_P(v), 40 \log n)$ ;
17:     $U \leftarrow$  add  $\text{pred}_P(v)$ ;
18:     $q \leftarrow$  Search  $N$  for  $q$  such that  $\text{half}_P(q) = \text{false}$  and  $\text{pred}_P(q) \notin U$ ;
19:     $P \leftarrow (p_{\text{start}}, \dots, \text{pred}_P(v)) + (q, \dots, p_{\text{end}}) + (v, \dots, \text{pred}_P(q))$ ;
20:     $p_{\text{end}} \leftarrow \text{pred}_P(q)$ ;
21: // If any of the “Search” parts of the algorithm fail, we abort the algorithm and return
    failure.

```

---

tree containing the remaining vertices) resp. concatenate two AVL trees in which the largest vertex in one tree is smaller than the smallest vertex in the other tree. It is well known that both of these operations can be done for AVL trees in  $\mathcal{O}(\log n)$  time, cf. Lemma 10 in Section 3 for more details.

► **Proposition 6.** *Applying the procedure  $\text{AddSingleCycle}$  at most  $2 \log n$  times will run in time  $\mathcal{O}(n)$  with high probability.*

**Proof.** We want to bound the number of Posa rotations we need to perform while we add at most  $2 \log n$  cycles. Each Posa rotation occurs at the end of a while loop in the pseudocode.

To incorporate a cycle we want to find a vertex  $v$  which, in the order of the path, is right after a vertex in  $N_{\text{start}}$  and is in the first half of  $P$ .  $P$  has size at least  $3n/4$  so the number of vertices in the first half is at least  $n/4$ . A random vertex therefore has a chance of at least  $1/4$  to be in the first half of  $P$ . So every vertex in  $\text{newneighbor}(c_{\text{start}}, 40 \log n)$  has probability at least  $1/4$  independently of being in the first half of  $P$  and different from the other vertices. This implies that the number of vertices in  $N_{\text{start}}$  which are also in the first half of  $P$  dominates a binomial distributed random variable  $F \sim \text{Bin}(40 \log n, 1/4)$ . For  $F$  we know the expectation to be  $10 \log n$  and by a Chernoff bound (11) the probability that  $F$  is less than  $\log n$  is  $\mathcal{O}(n^{-2})$ . We observe that where the Posa rotation happens is independent of  $N_{\text{start}}$ . So we apply a union bound that on fixed  $\mathcal{O}(n)$  many rotations of  $P$  the probability that there are less than  $\log n$  vertices of  $N_{\text{start}}$  in the first half of  $P$  is in  $\mathcal{O}(n^{-1})$ . This implies that any call to  $\text{newneighbor}(p_{\text{end}})$  has a chance of at least  $\log n/n$  to be right after a vertex in  $N_{\text{start}}$  and also in the first half of  $P$ . As each call to  $\text{newneighbor}()$  is independent, the number of tries we must make is geometrically distributed with success probability  $\log n/n$  and we must succeed at most  $2 \log n$  many times. This means the number of Posa rotations

is dominated by a negative binomial distributed random variable  $R \sim NB(2 \log n, \log n/n)$ . So by the concentration of the negative binomial distribution (Lemma 14) the probability that we need to try more than  $4n$  times is at most  $\mathcal{O}(\log^{-1} n)$ . Before every Posa rotation we try `newneighbor`( $p_{end}, 40 \log n$ ) so  $40 \log n$  tries. This proves an upper bound on the number of Posa rotations of  $\mathcal{O}(n/\log n)$  with high probability.

**Posa rotation.** We summarize the operations we need to do per Posa rotation. This assumes that we already failed to find  $v$  which is both after a vertex in  $N_{start}$  and also in the first half of  $P$ . We expose  $40 \log n$  new neighbors of  $p_{end}$  and  $40 \log n$  of the vertex before  $v$  on the path, we need to Posa rotate by splitting the path twice and then joining twice. Checking whether a vertex is in  $U$  and adding vertices to  $U$  is a constant time operation with a lookup table. All of these operations by choice of proper datastructure (Lemma 8 and 10) are done in  $\mathcal{O}(\log n)$ . So over all Posa rotations these sum up to a runtime of at most  $\mathcal{O}(n)$ . Additionally we need to find the vertex  $v$  in the first half of  $P$  with  $pred_P(v) \notin U$ . Since  $U$  is much smaller than  $n/8$  and  $|P| \geq 3n/4$  the number of possible vertices is at least  $n/4$ . This means that if we test a random vertex, the probability that `half_P`() returns true and its predecessor is not in  $U$  is at least  $1/4$ . So the number times we need to call `half_P`() is dominated by a geometric distribution with success probability  $1/4$ . Similarly to find the vertex  $q$  in the second half of  $P$  with  $pred_P(q) \notin U$ , the number of times we need to call `half_P`() is also dominated by a geometric distribution with success probability  $1/4$ . So over all rotations, the number of times we need to call `half_P`() is dominated by a negative binomial distribution  $H \sim NB(2 \cdot \mathcal{O}(n/\log n), 1/4)$ . So by the concentration of the negative binomial distribution (Lemma 14) the probability that we need to call `half_P` more than  $\mathcal{O}(n/\log n)$  times is  $\mathcal{O}(\log n/n)$ . And since we can perform `half_P`() in time  $\mathcal{O}(\log n)$  by Lemma 10 these have a total runtime of  $\mathcal{O}(n)$  with high probability.

**Incorporating cycles.** Very similarly we bound the time we need to incorporate the cycles. To find the vertex  $v$  which in the order of the path is right after a vertex in  $N_{start}$  and is in the first half of  $P$  we need to call `half_P` until we succeed. Note that since  $|N_{start}| \leq 40 \log n$  and as we proved above at least  $\log n$  vertices of them are in the first half of  $P$ , every call to `half_P`() from a random vertex after a vertex in  $N_{start}$  has a chance of succeeding of at least  $1/40$ . This means the number of times we call `half_P` is again dominated by a negative binomial distribution  $NB(2 \log n, 1/40)$  and this runtime is negligible with high probability. As we only incorporate a cycle  $2 \log n$  times, also the join and split operations as well as the exposing of  $N_{end}$  and searching for  $q$  are negligible compared to the  $\mathcal{O}(n)$  runtime.

Note also that we only call `newneighbor`() of vertices we then add to  $U$  and then not again during the entire algorithm so no vertex has `newneighbor`() called more than  $40 \log n$  times. At most  $\mathcal{O}(n/\log n)$  many vertices are added to  $U$ , and  $U$  is small enough so that it is always much smaller than  $n/8$ .

This concludes the proof of Proposition 6. ◀

► **Lemma 7.** *Given a Hamilton path we can transform it to a Hamilton cycle in  $\mathcal{O}(n)$  time.*

**Proof.** Calling the algorithm `AddSingleCycle` with the cycle being  $p_{start}$ , but instead looking for  $v$  such that a vertex after  $v$  is in the neighborhood of  $p_{start}$  instead of a predecessor gives us a cycle  $C = (p_{start}, \dots, v) + (p_{end}, \dots, after_P(v))$ . Analysis of runtime equivalent to the analysis of `AddSingleCycle`. ◀

Propositions 4 and 6 as well as Lemma 7 show that all components of the algorithm run in time  $\mathcal{O}(n)$ . It is also easy to check that both phases together require at most  $50 \log n$  calls to `newneighbor`() from any fixed vertex, so the assumptions of Lemma 8 do hold. So choosing  $C$  large enough, say  $C = 200$ , suffices to guarantee that with high probability the random graph is such that all vertices have more neighbors than we query. This thus concludes the proof of Theorem 1.

### 3 Datastructures

In this section we give the details of the data structures that we used within our algorithm.

#### 3.1 Querying a new vertex

As explained above, we assumed throughout the analysis of our algorithm that we have access to a function `newneighbor(v)`, that returns for a given vertex  $v$  a neighbor  $w$  that is *uniformly* distributed in  $V - v$  and whose result is *independent* from all previous calls.

► **Lemma 8** (`newneighbor`). *It is possible to interact with the graph  $G_{n,p}$ ,  $p \geq \frac{C \log n}{n}$ , with an algorithmic procedure `newneighbor(v)` which has the following properties with high probability:*

- (i) *Calling `newneighbor(v)` returns a neighbor of  $v$  distributed uniformly among  $V - v$  and independent of all calls so far – as long as we make at most  $\mathcal{O}(n)$  calls to `newneighbor()` altogether and every vertex is queried at most  $100 \log n$  times.*
- (ii) *The total run time of all  $\mathcal{O}(n)$  calls is  $\mathcal{O}(n)$ .*

**Proof.** To realize such a function `newneighbor()`, it is important to make the adjacency lists independent of each other. To realize this we transform the graph  $G$  (which is distributed as a random graph  $G_{n,p}$ ) into a directed graph  $G'$  distributed as  $D_{n,p/2}$  (in which each directed edge is present independently with probability  $p/2$ ). It is well known how this can be done. In particular, we can sample  $D_{n,p/2}$  from  $G_{n,p}$  as a subgraph such that every edge in the directed graph is also an undirected edge in the  $G_{n,p}$ . More precisely, we do the following for every edge  $\{i, j\}$  of  $G$ : with probability

$$\begin{aligned} \frac{1}{2} - \frac{p}{4} & \text{ set } (i, j) \in G' \text{ and } (j, i) \notin G' \\ \frac{1}{2} - \frac{p}{4} & \text{ set } (i, j) \notin G' \text{ and } (j, i) \in G' \\ \frac{p}{4} & \text{ set } (i, j) \in G' \text{ and } (j, i) \in G' \\ \frac{p}{4} & \text{ set } (i, j) \notin G' \text{ and } (j, i) \notin G' \end{aligned}$$

In order to be consistent with the transformation from  $G$  to  $G'$  and to not lose too much time we only direct the edges once we see it for the first time. To recall the made decision, we store the random choices of the edges that we have encountered so far into a hashtable. Thus, we can check for each edge that we obtain from querying the adjacency list of a vertex in  $G$ , whether we have seen this edge already and if so, which orientation we have chosen. The hash table has size  $n$  and we use a hashfunction which is 4-wise independent. This way the variance of the number of collisions is equal to a random function, and therefore the time we need for hashing is  $\mathcal{O}(n) + \mathcal{O}(\text{number of collisions}) = \mathcal{O}(n)$ , which can be seen by applying Chebyshev's inequality. A more detailed argument of why linear probing with hash functions works in this context can be found in [24, 31].

Finally, we want the distribution of the next edge to be uniform among the vertices. For this we need to resample from the already seen edges. Assuming we have revealed  $d$  many edges from  $v$  we flip a biased coin. With probability  $d/(n-1)$  we retake an old neighbor and output it, one chosen uniformly at random, and with probability  $1 - d/(n-1)$  we take the next vertex in the adjacency list (which is also in  $D_{n,p}$ ). Otherwise, we return one of the previously seen neighbors uniformly at random. In this way any vertex has probability exactly  $1/(n-1)$  to be returned by `newneighbor(v)`. ◀

### 3.2 Expansion

What we need from the random graph are properties of good expansion. Given the adjacency list of a vertex  $v$  we define the  $d$ -neighborhood  $N_d(v) \subseteq V(G)$  to be the set of the first  $\lceil d \rceil$  calls to the function `newABneighbor(v)`. In the analysis of the algorithm we make use of the following lemma.

► **Lemma 9 (Neighborhood Lemma).** *Let  $G_{n/2, n/2, p}$  be a random bipartite graph with  $p \geq \frac{C \log(n)}{n}$  and partitions  $A$  and  $B$ . Then with high probability we have for all subsets  $A' \subseteq A$  that the  $d$ -neighborhood of  $A'$  is of size at least*

$$|N_{d(A')}(A')| \geq \frac{1}{100} |A'| \cdot d(A'), \quad (1)$$

where  $d(A') = \min(\sqrt{\frac{n}{|A'|}}, \log(n))$ .

**Proof.** The proof follows from a straight forward calculation of probabilities. Let us assume by contradiction there exists a set  $A' \subseteq A$  with  $|N_{d(A')}(A')| < \frac{1}{100} |A'| \cdot d(A')$ . Then there is a set  $B' \subseteq B$  of size  $|B'| = \frac{1}{100} |A'| \cdot d(A')$  containing this  $d$ -neighborhood,  $N_{d(A')}(A') \subseteq B'$ . So this is a probability we want to bound from above. The probability for a single vertex in  $A'$  to have its  $d$ -neighborhood contained in a fixed set  $B'$  is  $\left(\frac{|B'|}{n}\right)^{d(A')}$  since the  $d$ -neighborhood is  $d(A')$  vertices chosen from  $B$  uniformly and independently at random. The probability for two specific sets  $A' \subseteq A$  and  $B' \subseteq B$  to have this property is  $(|B'|/n)^{|A'| \cdot d(A')}$ . Now take the union bound over all possible sets  $A'$  and  $B'$  (with  $|B'| = \frac{1}{100} |A'| \cdot d(A')$ ):

$$Pr[(1) \text{ false}] \leq \sum_{A', B'} Pr[B' \text{ contains } N_{d(A')}(A')] = \sum_{i=1}^n \binom{n}{i} \binom{n}{\frac{1}{100} i d(i)} \left(\frac{\frac{1}{100} i \cdot d(i)}{n}\right)^{i \cdot d(i)}.$$

Then we apply an approximation for the binomial coefficients:  $\binom{n}{k} \leq \left(\frac{en}{k}\right)^k$ . We see that  $\frac{1}{4} \log \frac{100n}{i \cdot d(i)} \geq \log(e \cdot n/i)/d(i)$  so

$$Pr[(1) \text{ false}] \leq \sum_{i=1}^n \exp\left(i \cdot d(i) \cdot \left(-\frac{1}{2} \log\left(\frac{100n}{i \cdot d(i)}\right)\right)\right).$$

Now  $d(i)$  is a known function of  $i$ . So we distinguish between two cases. When  $i \geq n/\log^2(n)$ , then  $d(i) = \sqrt{n/i}$ . And we can calculate ( $i \leq n$ )

$$\sum_{i=1}^n \left(\frac{i^{1/4} \cdot n^{1/4}}{10 \cdot n^{1/2}}\right)^{\sqrt{n \cdot i}} \leq \sum_{i=1}^n \left(\frac{1}{10}\right)^{\sqrt{n \cdot i}} \leq \mathcal{O}(n^{-2}).$$

On the other hand if  $i \leq n/\log^2(n)$ , then  $d(i) = \log(n)$ . And we can calculate

$$\sum_{i=1}^{n/\log^2(n)} \left(\frac{i^{1/2} \cdot \log(n)^{1/2}}{10 \cdot n^{1/2}}\right)^{i \log(n)} \leq \sum_{i=1}^{n/\log^2(n)} \left(\frac{1}{10 \cdot \log(n)^{1/2}}\right)^{i \log(n)} \leq \mathcal{O}(n^{-2}).$$

Together this implies that the lemma holds for random graphs with probability  $\geq 1 - \mathcal{O}(n^{-2})$ . ◀

### 3.3 AVL Trees

► **Lemma 10** (AVL Trees). *We can store a path (or cycle) in an AVL tree joint with a linked list datastructure and can perform the following operations (where we view the cycle as a path split at an arbitrary point):*

- *For any vertex  $v$  find the vertex preceding or succeeding it in the path in constant time  $\mathcal{O}(1)$*
- *For any vertex  $v$  searching the path it is in and determining whether it is in the first or second half of it in time  $\mathcal{O}(\log n)$*
- *Split the path into two paths in time  $\mathcal{O}(\log n)$*
- *Concatenate two paths into one by adding the endpoint of one to the start of the other in time  $\mathcal{O}(\log n)$*

**Proof.** We combine an AVL tree, which is a balanced binary search tree, with a linked list. The AVL tree is built on the order sequence of the path as if numbering the vertices along the path from 1 to  $|P|$ . The linked list ensures that going forward and backward on the path is done in constant time, where the AVL tree can perform search (for the half function) in  $\mathcal{O}(\log n)$ . A split of the path is nothing other than splitting the AVL tree at a leaf node into two trees such that all the nodes smaller go into one tree and all the nodes larger go into the other. The concatenate is the inverse of the split and only requires attaching the smaller tree to the larger one at the appropriate node and rebalancing up to the root. Both operations run in  $\mathcal{O}(\log n)$  time. AVL trees are by now a part of basic datastructure lectures and in particular the split and concatenate operations can be found e.g. in the book by Knuth [18] see page 473, which also cites from [10] or more generally on AVL trees see [25]. ◀

## 4 Concluding remarks

In this paper we presented a simple randomized algorithm based on iterative random walks that construct a Hamilton cycle in time linear in the number of vertices. Our algorithm is based on first building (two) random perfect matchings. The key idea here is to expose more and more edges of the currently unmatched vertices, where the exact number of these exposed neighbors is a function of the currently unmatched vertices. Our analysis requires that the density of the random graph  $G_{n,p}$  is at least  $p \geq \frac{C \log n}{n}$ .  $C$  is chosen such that we have a sufficient minimum degree with high probability.

We leave it as an open question whether our approach can be modified to also find Hamilton cycles in  $G_{n,p}$  for  $p$  at the threshold for existence of Hamilton cycles. This certainly requires additional ideas.

---

### References

- 1 Miklós Ajtai, János Komlós, and Endre Szemerédi. First occurrence of Hamilton cycles in random graphs. *North-Holland Mathematics Studies*, 115(C):173–178, 1985.
- 2 Peter Allen, Julia Böttcher, Yoshiharu Kohayakawa, and Yury Person. Tight hamilton cycles in random hypergraphs. *Random Structures & Algorithms*, 46(3):446–465, 2015.
- 3 Yahav Alon and Michael Krivelevich. Finding a Hamilton cycle fast on average using rotations and extensions. *Random Structures & Algorithms*, 57(1):32–46, 2020.
- 4 Dana Angluin and Leslie G Valiant. Fast probabilistic algorithms for Hamiltonian circuits and matchings. *Journal of Computer and System Sciences*, 18(2):155–193, 1979.
- 5 Richard Arratia, Andrew D Barbour, and Simon Tavaré. *Logarithmic combinatorial structures: a probabilistic approach*, volume 1. European Mathematical Society, 2003.

- 6 Richard Arratia and Simon Tavaré. The cycle structure of random permutations. *The Annals of Probability*, pages 1567–1591, 1992.
- 7 Tom Bohman and Alan Frieze. Hamilton cycles in 3-out. *Random Structures & Algorithms*, 35(4):393–417, 2009.
- 8 Bela Bollobas, Trevor I. Fenner, and Alan M. Frieze. An algorithm for finding Hamilton paths and cycles in random graphs. *Combinatorica*, 7(4):327–341, 1987.
- 9 Bela Bollobás, Trevor I. Fenner, and Alan M. Frieze. Hamilton cycles in random graphs with minimal degree at least  $k$ . *A tribute to Paul Erdos, edited by A. Baker, B. Bollobás and A. Hajnal*, pages 59–96, 1990.
- 10 C.A. Crane. *Linear Lists and Priority Queues as Balanced Binary Trees*. Computer Science Department. Department of Computer Science, Stanford University., 1972.
- 11 Trevor I Fenner and Alan M Frieze. Hamiltonian cycles in random regular graphs. *Journal of Combinatorial Theory, Series B*, 37(2):103–112, 1984.
- 12 Asaf Ferber, Michael Krivelevich, Benny Sudakov, and Pedro Vieira. Finding hamilton cycles in random graphs with few queries. *Random Structures & Algorithms*, 49(4):635–668, 2016.
- 13 Alan Frieze. Hamilton cycles in random graphs: a bibliography. *arXiv preprint*, 2019. [arXiv:1901.07139](https://arxiv.org/abs/1901.07139).
- 14 Alan Frieze, Mark Jerrum, Michael Molloy, Robert Robinson, and Nicholas Wormald. Generating and counting hamilton cycles in random regular graphs. *Journal of Algorithms*, 21(1):176–198, 1996.
- 15 Alan M Frieze. Finding hamilton cycles in sparse random graphs. *Journal of Combinatorial Theory, Series B*, 44(2):230–250, 1988.
- 16 Yuri Gurevich and Saharon Shelah. Expected computation time for Hamiltonian path problem. *SIAM Journal on Computing*, 16(3):486–502, 1987.
- 17 Svante Janson, Tomasz Luczak, and Andrzej Rucinski. *Random graphs*, volume 45. John Wiley & Sons, 2011.
- 18 Donald E Knuth. *The art of computer programming: Volume 3: Sorting and Searching*. Addison-Wesley Professional, 1998.
- 19 János Komlós and Endre Szemerédi. Limit distribution for the existence of Hamiltonian cycles in a random graph. *Discrete mathematics*, 43(1):55–63, 1983.
- 20 Aleksei Dmitrievich Korshunov. Solution of a problem of erdős and renyi on Hamiltonian cycles in nonoriented graphs. *Doklady Akademii Nauk*, 228(3):529–532, 1976.
- 21 Kenneth Maples, Ashkan Nikeghbali, and Dirk Zeindler. On the number of cycles in a random permutation. *Electron. Commun. Probab.*, 17:13 pp., 2012.
- 22 Richard Montgomery. Hamiltonicity in random graphs is born resilient. *Journal of Combinatorial Theory, Series B*, 139:316–341, 2019.
- 23 Rajko Nenadov, Angelika Steger, and Miloš Trujić. Resilience of perfect matchings and hamiltonicity in random graph processes. *Random Structures & Algorithms*, 54(4):797–819, 2019.
- 24 Anna Pagh, Rasmus Pagh, and Milan Ruzic. Linear probing with constant independence. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 318–327, 2007.
- 25 Ben Pfaff. An introduction to binary search trees and balanced trees. *Libavl Binary Search Tree Library*, 1:19–20, 1998.
- 26 Robert W. Robinson and Nicholas C. Wormald. Almost all regular graphs are Hamiltonian. *Random Structures & Algorithms*, 5(2):363–374, 1994.
- 27 Ronitt Rubinfeld and Asaf Shapira. Sublinear time algorithms. *SIAM Journal on Discrete Mathematics*, 25(4):1562–1588, 2011.
- 28 Eli Shamir. How many random edges make a graph Hamiltonian? *Combinatorica*, 3(1):123–131, 1983.
- 29 Benny Sudakov and Van H Vu. Local resilience of graphs. *Random Structures & Algorithms*, 33(4):409–433, 2008.

- 30 Andrew Thomason. A simple linear expected time algorithm for finding a hamilton path. *Discrete Mathematics*, 75(1-3):373–379, 1989.
- 31 Mikkel Thorup and Yin Zhang. Tabulation based 4-universal hashing with applications to second moment estimation. In *SODA*, volume 4, pages 615–624, 2004.

## A

 Concentration Inequalities

We mention here some well-known inequalities used to show concentration on random variables. By the concentration of a random variable  $X$  we usually mean there exist constants  $c$  and  $C$  such that with high probability  $c\mathbb{E}[X] \leq X \leq C\mathbb{E}[X]$ . Often also we ask  $C$  to be 2. Although these are by no means new insights, we present them here for completion and as a help for the reader. The Chernoff and Chebyshev inequality can be found e.g. in the book [17].

► **Theorem 11** (Chernoff Inequality). *If  $X$  is distributed as a binomial random variable  $X \sim \text{Bin}(n, p)$  and  $0 < \varepsilon \leq 3/2$ , then*

$$\Pr[|X - \mathbb{E}[X]| \geq \varepsilon \mathbb{E}[X]] \leq 2e^{-\frac{\varepsilon^2 \mathbb{E}[X]}{3}}.$$

► **Theorem 12** (Chebyshev Inequality). *For any random variable  $X$  for which the variance  $\text{Var}[X]$  exists,*

$$\Pr[|X - \mathbb{E}[X]| \geq t] \leq \frac{\text{Var}[X]}{t^2}.$$

We use the term negative binomial distribution in the analysis and since this is defined slightly differently sometimes we give here the definition we use.

► **Definition 13.** *Let  $X_i$  be independent bernoulli random variables with probability of being one is  $p$  for any  $i \in \mathbb{N}$ . For any  $r \in \mathbb{N}$  let  $Y$  be index of the  $r$ -th  $X_i$  which evaluates to 1. Then  $Y$  has a negative binomial distribution  $NB(r, p)$ .*

We observe that a negative binomial distribution  $Y \sim NB(r, p)$  is equivalent to  $Y$  being distributed as the sum of  $r$  geometric random variables with success probability  $p$ . Further a simple corollary from the Chebyshev inequality:

► **Corollary 14.** *For a negative binomial distributed variable  $Y \sim NB(r, p)$*

$$\Pr\left[Y \geq 2\frac{r}{p}\right] \leq \frac{1}{r}.$$

**Proof.** We calculate  $\mathbb{E}[Y] = r/p$  and  $\text{Var}[Y] = r(1-p)/p^2$  and apply Chebyshev.

$$\Pr\left[Y \geq 2\frac{r}{p}\right] \leq \Pr\left[|Y - \mathbb{E}[Y]| \geq \frac{r}{p}\right] \stackrel{\text{Chebyshev}}{\leq} \frac{1-p}{r} \leq \frac{1}{r} \quad \blacktriangleleft$$





# Communication Memento: Memoryless Communication Complexity

Srinivasan Arunachalam

IBM T. J. Watson Research Center, Yorktown Heights, NY, USA  
srinivasan.arunachalam@ibm.com

Supartha Podder

University of Ottawa, Canada  
spodder@uottawa.ca

---

## Abstract

We study the communication complexity of computing functions  $F : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$  in the *memoryless communication model*. Here, Alice is given  $x \in \{0, 1\}^n$ , Bob is given  $y \in \{0, 1\}^n$  and their goal is to compute  $F(x, y)$  subject to the following constraint: at every round, Alice receives a message from Bob and her reply to Bob solely depends on the message received and her input  $x$  (in particular, her reply is independent of the information from the previous rounds); the same applies to Bob. The cost of computing  $F$  in this model is the *maximum* number of bits exchanged in any round between Alice and Bob (on the worst case input  $x, y$ ). In this paper, we also consider variants of our memoryless model wherein one party is allowed to have memory, the parties are allowed to communicate quantum bits, only one player is allowed to send messages. We show that some of these different variants of our memoryless communication model capture the garden-hose model of computation by Buhrman et al. (ITCS'13), space-bounded communication complexity by Brody et al. (ITCS'13) and the overlay communication complexity by Papakonstantinou et al. (CCC'14). Thus the memoryless communication complexity model provides a unified framework to study all these space-bounded communication complexity models.

We establish the following main results: (1) We show that the memoryless communication complexity of  $F$  equals the logarithm of the size of the smallest bipartite branching program computing  $F$  (up to a factor 2); (2) We show that memoryless communication complexity equals garden-hose model of computation; (3) We exhibit various exponential separations between these memoryless communication models.

We end with an intriguing open question: can we find an explicit function  $F$  and universal constant  $c > 1$  for which the memoryless communication complexity is at least  $c \log n$ ? Note that  $c \geq 2 + \epsilon$  would imply a  $\Omega(n^{2+\epsilon})$  lower bound for general formula size, improving upon the best lower bound by Nečiporuk [33].

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Communication complexity

**Keywords and phrases** Communication complexity, space complexity, branching programs, garden-hose model, quantum computing

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.61

**Related Version** A full version of the paper is available at [2], <https://arxiv.org/abs/2005.04068>.

**Funding** *Srinivasan Arunachalam*: Supported in part by the Army Research Laboratory and the Army Research Office under grant number W1975117.

**Acknowledgements** We thank Florian Speelman for the proof of the first inequality of Theorem 21 and letting us include it in our paper. We would like to thank Mika Göös, Robert Robere, Dave Touchette and Henry Yuen for helpful pointers. SP also thanks Anne Broadbent and Alexander Kerzner for discussions during the early phase of this project as a part of discussing gardenhose model. We also thank anonymous reviewers for their helpful comments and positive feedback. Part of this work was done when SA was visiting University of Ottawa (hosted by Anne Broadbent) and University of Toronto (hosted by Henry Yuen) and thank them for their hospitality.



© Srinivasan Arunachalam and Supartha Podder;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 61; pp. 61:1–61:20



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

## 1 Introduction

Yao [44] introduced the model of *communication complexity* in 1979 and ever since its introduction, communication complexity has played a pivotal role in understanding various problems in theoretical computer science. In its most general form in this model, the goal is the following: there are two separated parties usually referred to as Alice and Bob, Alice is given an  $n$ -bit string  $x \in \{0, 1\}^n$  and similarly Bob is given  $y \in \{0, 1\}^n$  and together they want to compute  $F(x, y)$  where  $F : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$  is a function known to both of them. Here Alice and Bob are given unlimited computational time and memory and the *cost* of any communication protocol between Alice and Bob is the *total number of bits exchanged* between them. Clearly a trivial protocol is Alice sends her input  $x$  to Bob who can then compute  $F(x, y)$ , which takes  $n$  bits of communication. Naturally, the goal in communication complexity is to minimize the number of bits of communication between them before computing  $F(x, y)$ . The *deterministic communication complexity* of a function  $F$  (denoted  $D(F)$ ) is defined as the total number of bits of communication before they can decide  $F(x, y)$  on the worst-case inputs  $x, y$ .

Since its introduction there have been various works that have extended the standard deterministic communication model to the setting where Alice and Bob are allowed to share randomness and need to output  $F(x, y)$  with high probability (probability taken over the randomness in the protocol). Apart from this there have been studies on non-deterministic communication complexity [42], quantum communication complexity [43] (wherein Alice and Bob are allowed to share *quantum bits* and possibly have shared entanglement), *unbounded error communication complexity* [36] and their variants. One-way variants have also been considered where only Alice sends messages to Bob. Study of these different models of communication complexity and their variants have provided many important results in the fields of VLSI [34], circuit lower bounds [22], algorithms [1], data structures [32], property testing [7], streaming algorithms [6], computational complexity [8], extended formulations [18].<sup>1</sup>

### 1.1 Background

**Space-bounded communication complexity.** In the context of our current understanding of computation, the study of space required to solve a problem is a central topic in complexity theory. Several space-bounded models such as width-bounded branching programs [28], limited depth circuits, straight line protocols [29] have been widely studied in this context. In this direction variants of communication complexity have also been analyzed to better understand communication-space trade-offs [23, 26, 28]. In particular, the relation between space-bounded computation and communication complexity was formally initiated by Brody et al. [11] who considered the following question: what happens if we change the standard communication model such that, in each step of communication, Alice and Bob are limited in their ability to store the information from the previous rounds (which includes their private memory and messages exchanged). In this direction, they introduced a new model wherein Alice and Bob each are allowed to store at most  $s(n)$  bits of memory and showed that unlike the standard communication complexity, in this model *super-linear* lower bounds on the amount of communication is possible.<sup>2</sup> Brody et al. mainly studied the one-way communication complexity variant of this limited memory model in which Bob can have two

<sup>1</sup> For more on communication complexity and its applications, we refer the interested reader to the standard textbooks for communication complexity [27, 31].

<sup>2</sup> We remark that the separations obtained by [11] were for non-Boolean functions.

types of memory: an oblivious memory (depends only on Alice's message) and a non-oblivious memory (for computation). With these definitions, they obtained memory hierarchy theorems for such communication models analogous to the space hierarchy theorem in the Turing machine world.

**Overlay communication complexity.** Subsequently, Papakonstantinou, et al. [35] defined a similar space-bounded *one-way* communication model wherein Alice has unlimited memory and Bob has either no memory or constant-sized memory. At each round, messages from Alice to Bob consist of at most  $t(n)$  bits and the complexity of computing  $F$  is the maximum  $t(n)$  required over all inputs to  $F$ . They characterized the complexity in this model by an elegant combinatorial object called the *rectangle overlay* (which is defined in Section 4.2). They also managed to establish connections between their model and the well-known communication complexity polynomial hierarchy, introduced by Babai, Frankl and Simon [3]. Papakonstantinou et al. [35] showed that the message length in their model corresponds to the oblivious memory in a variant of space-bounded model, introduced by Brody et al. [11], where Bob only has access to an oblivious memory.

**Garden-hose model.** Another seemingly unrelated complexity model, the garden-hose complexity was introduced by Buhrman et al. [15] to understand quantum attacks on position-based cryptographic schemes (see Section 5.1 for a formal definition). Polynomial size garden-hose complexity is known to be equivalent to Turing machine log-space computations with pre-processing. In the garden-hose model two distributed players Alice and Bob use several pipes to send water back and forth and compute Boolean functions based on whose side the water spills. Garden-hose model was shown to have many connections to well-established complexity models like formulas, branching programs and circuits. A long-standing open question in this area is, is there an explicit function on  $n$  bits whose garden-hose complexity is super-linear in  $n$ ?

**Branching programs.** Another unrelated computation model is the branching program. Understanding the size of De Morgan formulas that compute Boolean functions has a long history. In particular, there has been tremendous research in understanding lower bounds on size of De Morgan formulas computing a Boolean function. Similar to formulas, branching programs have also been well-studied in complexity theory. For both branching programs and formulas, we have explicit functions which achieve quadratic (in input size) lower bounds on the size of the branching program/formula computing them. A few years ago, Tal [40] considered *bipartite formulas* for  $F : X \times Y \rightarrow \{0, 1\}$  (where each internal node computes an arbitrary function on either  $X$  or  $Y$ , but not both) and showed that the inner product function requires quadratic-sized formulas to compute. In the same spirit as Tal's result, a natural open question is, is there an explicit bipartite function which requires super-linear sized bipartite branching programs to compute?

Given these different models of computation, all exploring the effects on computation under various restrictions, a natural question is, can we view all of them in a unified way:

*Is there a model of communication that captures all the above computational models?*

**In this work** we introduce a very simple and new framework called *the memoryless communication complexity* which captures all the computational models mentioned above.

## 1.2 Memoryless Communication Models

We introduce a *natural* model of communication complexity which we call *the memoryless communication complexity*. Here, like the standard communication complexity, there are two parties Alice and Bob given  $x, y$  respectively and they need to compute  $F(x, y)$ , where  $F : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$  is known to both of them. However, we tweak the standard communication model in the following two ways: The first change is that Alice is “*memoryless*”, i.e., at every round Alice computes the next message to send solely based on only her input  $x$  and the message received from Bob in this round. She does not remember the entire transcript of messages that were communicated in the previous rounds and also forgets all the private computation she did in the previous rounds. Similarly Bob computes a message which he sends to Alice, based *only* on his input  $y$  and the message received from Alice in the current round. After Bob sends his message, he also forgets the message received and all his private computations. Alice and Bob repeat this procedure for a certain number of rounds before one of them outputs  $F(x, y)$ .

The second crucial change in the memoryless communication model is that the *cost* of computing  $F$  in this model is the *size of the largest message* communicated between Alice and Bob in any round of the protocol (here size refers to the number of bits in the message). Intuitively, we are interested in knowing what is the size of a re-writable message register (passed back and forth between Alice and Bob) sufficient to compute a function  $F$  on all inputs  $x$  and  $y$ , wherein Alice and Bob do not have any additional memory to remember information between rounds. We denote the *memoryless communication cost of computing*  $F$  as  $\text{NM}(F)$  (where  $\text{NM}$  stands for “no-memory”). We believe this communication model is very natural and as far as we are aware this memoryless communication model wasn’t defined and studied before in the classical literature.

Being more formal, we say  $F : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$  can be computed in the memoryless communication model with complexity  $t$ , if the following is true. For every  $x, y \in \{0, 1\}^n$  there exists functions  $\{f_x, g_y : \{0, 1\}^t \rightarrow \{0, 1\}^t\}$  such that, on input  $x, y$ , Alice and Bob use  $f_x$  and  $g_y$  respectively to run the following protocol: the first message in the protocol is  $f_x(0^t)$  from Alice to Bob and thereafter, for every message  $m_B$  Bob receives, he replies with deterministic  $m' = g_y(m_B)$  and similarly for every message  $m_A$  Alice receives she replies with  $m'' = f_x(m_A)$ . The protocol terminates when the transcript is  $(1^{t-1}b)$  at which point they output  $b$  as their guess for  $F(x, y)$ ; and we say the protocol computes  $F$  if for every  $x, y$ , the output  $b$  equals  $F(x, y)$ .  $\text{NM}(F)$  is defined as the smallest  $t$  that suffices to compute  $F$  (using the protocol above) for every  $x, y \in \{0, 1\}^n$ .

It is worth noting that in the memoryless communication protocol, Alice and Bob do not even have access to clocks and hence cannot tell in which round they are in (without looking at the message register). Hence, every memoryless protocol can be viewed as Alice and Bob applying *deterministic* functions (depending on their inputs) which map incoming messages to out-going messages. Also note that unlike the standard communication complexity, where a single bit-message register suffices for computing all functions (since everyone has memory), in the  $\text{NM}$  model because of the memoryless-ness we need more than a single bit register for computing almost all functions.

For better understanding, let us look at a protocol for the standard *equality* function defined as  $\text{EQ}_n : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$  where  $\text{EQ}_n(x, y) = 1$  if and only if  $x = y$ . It is well-known that  $\text{D}(\text{EQ}_n) = n$ . In our model, we show that  $\text{NM}(\text{EQ}_n) \leq \log n + 1$ : for  $i = 1, \dots, n$ , at the  $i$ th round, Alice sends the  $(\log n + 1)$ -bit message  $(i, x_i)$  and Bob returns

$(i, [x_i = y_i])$ ,<sup>3</sup> Alice increments  $i$  and repeats this protocol for  $n$  rounds. In case Bob finds an  $i$  for which  $x_i \neq y_i$ , he outputs 0, if not after  $n$  rounds they output 1. Note that this protocol didn't require Alice and Bob to have any memory and the length of the longest message in this protocol was  $\log n + 1$ . We discuss more protocols later in the paper and formally describe the memoryless communication model in Section 3.

**Variants of the memoryless model.** Apart from the memoryless communication complexity, we will also look at the “memory-nomemory communication complexity” wherein Alice is allowed to have memory (i.e., Alice can know which round she is in, can remember the entire transcript and her private computations of each round) whereas Bob doesn't have any memory during the protocol. The goal of the players remains to compute a function  $F$  and the *cost* of these protocols (denoted by  $M(F)$ ) is still defined as the smallest size of a message register required between them on the worst inputs. Apart from this, we will also consider the quantum analogues of these two communication models wherein the only difference is that Alice and Bob are allowed to send *quantum* bits. We formally describe these models of communication in Section 3. In order to aid the reader we first set up some notation which we use to describe our results: for  $F : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$ , let

1.  $NM(F)$  be the memoryless communication complexity of computing  $F$  wherein Alice and Bob both do not have any memory.
2.  $M(F)$  be the memory-nomemory communication complexity of computing  $F$  where Alice has memory and Bob doesn't have memory

Apart from these, we will also allow quantum bits of communication between Alice and Bob and the complexities in these models are denoted by  $QNM(F)$  and  $QM(F)$ . Additionally, we will consider the one-way communication variants wherein only Alice can send messages to Bob and the complexities in these models are denoted by  $NM^\rightarrow(F)$ ,  $M^\rightarrow(F)$ ,  $QNM^\rightarrow(F)$ ,  $QM^\rightarrow(F)$ .

### 1.3 Our Contributions

The main contribution in this paper is to first define the model of the memoryless communication complexity and consider various variants of this model (only some of which were looked at before in the literature). We emphasize that we view our main contribution as a new *simple* communication model that provides a conceptual – rather than technical – contribution to studying space complexity, bipartite branching programs and garden-hose complexity under a single model. Given the vast amount of research in the field of communication complexity, we believe that our memoryless model is a very natural model of computation. We now state of various connections between our memoryless communication model and other computational models.

**1. Characterization in terms of branching programs.** It is well-known that standard models of communication complexity are characterized by the so-called (monochromatic) “rectangles” that partition the communication matrix of the function Alice and Bob are trying to compute. In the study of the memoryless model, Papakonstantinou et al. [35] specifically consider the memory-nomemory model of communication complexity wherein Alice has a memory and Bob doesn't and they are restricted to one-way communication from Alice to Bob. They show a beautiful combinatorial rectangle-overlay characterization (denoted

---

<sup>3</sup> Here  $[\cdot]$  is the indicator of an event in the parenthesis.

$\text{RO}(F)$ ) of the memory-no memory communication model.<sup>4</sup> One natural idea is to improve the  $\text{RO}(F)$  complexity to a more fine-grained rectangle measure that could potentially also characterize  $\text{NM}(F)$ , but this doesn't seem to be true. The fact that both Alice and Bob do not have memory, doesn't allow them to “narrow” down into a rectangle allowing them to compute the function, instead they narrow down into a set of rectangles. This motivates the question, is there a natural characterization of even the memoryless communication model, in which both Alice and Bob do not have memory? Here we answer this in the positive. We provide a characterization of memoryless communication complexity using branching programs. In particular, we show that for every  $F : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$ , the memoryless complexity  $\text{NM}(F)$  is (up to a factor 2) equal to the logarithm of the size of the smallest bipartite branching program computing  $F$ .<sup>5</sup> We give a proof of this statement in Theorem 14.

**2. Characterization in terms of garden-hose complexity.** The garden-hose model of computation was introduced by Buhrman et al. [15] to understand quantum attacks on position-based cryptographic schemes. It is a playful communication model where two players compute a function with set of pipes, hoses and water going back-and-forth through them. Alice and Bob start with  $s$  pipes and based on their private inputs “match” some of the openings of the pipes on their respective sides. Alice also connects a water tap to one of the open pipes. Then based on which side the water spills they decide on the function value. Naturally they want to minimize the number of pipes required over all possible inputs and the garden-hose complexity  $\text{GH}(F)$  is defined to be the minimum *number of pipes* required to compute  $F$  this way. Given its puzzle-like structure, there have been several works to understand this model and various connections between the garden-hose model and other branches of theoretical computer science were established [15, 25, 39, 14, 16, 38, 17].

On the other hand, space-bounded communication complexity was introduced by Brody et al. [11] to study the effects on communication complexity when they players are limited in their ability to store information from previous rounds. Here Alice and Bob each have at most  $s(n)$  bits of memory. Based on their private inputs  $x, y$  they want to compute the function in a manner in which at each round Alice receives a single bit message from Bob and based on her input  $x$ , the incoming message  $m_B$  and her previous  $s(n)$ -bit register content, she computes a new  $s(n)$ -bit register and decides whether to stop and output 0/1 or to continue. Bob does the same. space-bounded communication complexity  $\text{SM}(F)$  of computing a function  $F$  is the minimum register size  $s(n)$  required to compute the function on the hardest input.

It was already shown by [11] that for every function, the logarithm of the garden-hose complexity and the space-bounded communication complexity is factor 2 related. It is also easy to show that our newly defined memoryless communication complexity is also factor 2 related to the space-bounded communication complexity by [11]:  $\text{NM}(F) \leq 2 \cdot \text{SM}(F) + 1$ , and  $\text{SM}(F) \leq \text{NM}(F) + \log \text{NM}(F)$ . We give a proof of this statement in Lemma 20. Thus it immediately follows that the logarithm of the garden-hose complexity and the memoryless communication complexity of any function is also at most factor 3 related. However we improve this relation using an elegant trick of [30] that allows one to make computations reversible; and thereby show that for every function  $F$ ,  $\text{NM}(F)$  and  $\text{GH}(F)$  are equal up to an additive term 4.

$$\log \text{GH}(F) - 4 \leq \text{NM}(F) \leq \log \text{GH}(F).$$

<sup>4</sup> This rectangle-overlay complexity is formally defined in Section 4.2.

<sup>5</sup> We defer the formal definition of such branching programs to Section 2 and Section 4.2.



We give a proof of this in Theorem 21. Hence, the memoryless communication complexity model provides a clean framework for studying all these apparently different looking computational models.

As an immediate application of this new characterization of the garden-hose model, we get a better upper bound for the garden-hose complexity of the indirect storage access function.

► **Definition 1** (Indirect Storage Access). *Let  $n \geq 4$  and  $m \geq 2$  be such that  $n = 2m + \log(m/\log m)$ . The Indirect storage access function  $\text{ISA}_n : \{0,1\}^n \rightarrow \{0,1\}$  is defined on the input string  $(\vec{x}, \vec{y}_1, \dots, \vec{y}_{2^k}, \vec{z})$  where  $k = \log\left(\frac{m}{\log m}\right)$ ,  $\vec{x} \in \{0,1\}^m$ ,  $\vec{y} \in \{0,1\}^{\log m}$ ,  $\vec{z} \in \{0,1\}^k$ . Then  $\text{ISA}_n(\vec{x}, \vec{y}_1, \dots, \vec{y}_{2^k}, \vec{z})$  is evaluated as follows: compute  $a = \text{Int}(z) \in [2^k]$ , then compute  $b = \text{Int}(\vec{y}_a) \in [m]$  and output  $x_b$ .*

*For the communication complexity version Alice gets  $\vec{x}, \vec{z}$  and Bob gets  $\vec{y}_1, \dots, \vec{y}_{2^k}$ , they want to compute  $\text{ISA}_n(\vec{x}, \vec{y}_1, \dots, \vec{y}_{2^k}, \vec{z})$ .*

It was conjectured in [25] that the Indirect Storage Access function has garden-hose complexity  $\Omega(n^2)$ . This function is known to have  $\Omega(n^2)$  lower bound for the branching program [41] and thus is believed to be hard for garden-hose model in general.<sup>6</sup> But it is easy to see that  $\text{NM}(\text{ISA}) \leq \log n$ : Alice sends  $\vec{z}$  to Bob who then replies with  $\vec{y}_{\text{Int}(z)}$ . Finally Alice computes the output. Thus using the memoryless-garden-hose equivalence (in Theorem 21) we immediately get  $\text{GH}(\text{ISA}) \leq 16n$  (thereby refuting the conjecture of [25]).

**3. Separating these models.** We then establish the following inequalities relating the various models of communication complexity.<sup>7</sup>

$$\begin{array}{ccccccc} \text{M}(F) & \leq & \text{NM}(F) & = & \log \text{GH}(F) & \leq & \text{M}^\rightarrow(F) \leq \text{NM}^\rightarrow(F) \\ \vee | & & * \vee | & & || & & \vee | \quad \vee | \\ \text{QM}(F) & & \text{QNM}(F) & & 2 \cdot \text{S}(F) & & \text{QM}^\rightarrow(F) \quad \text{QNM}^\rightarrow(F) \end{array}$$

Furthermore, except the inequality marked by  $(*)$ , we show the existence of various functions  $F : \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}$  for which every inequality is exponentially weak. In order to prove these exponential separations we use various variants of well-known functions such as inner product, disjointness, Boolean hidden matching problem, gap-hamming distance problem. Giving exponential separations between quantum and classical communication complexity<sup>8</sup> is an extensively studied subject [13, 12, 21, 4, 20] and in this paper we show such separations can also be obtained in the memoryless models. We provide the proof in Theorems 17 and 18.

In this paper, we are not been able to give a large separation between QNM and NM, primarily because all lower bound techniques we have for NM seem to apply for QNM as well, e.g., the deterministic one-way communication complexity and the non-deterministic communication complexity<sup>9</sup>. The only “trivial” separation we can give is a factor-2 gap

<sup>6</sup> In a typical garden-hose protocol for computing  $\text{ISA}_n$ , Alice uses  $m$  pipes to describe  $\vec{z}$  to Bob (each pipe for a different value of  $\vec{z}$ ). Bob can then use another set of  $m$  pipes to send  $\vec{y}_{\text{Int}(z)}$  to Alice. But since  $\vec{y}_i$ s need to be unique it appears that Bob needs  $m$  set of such  $m$  pipes in the worst case. This many-to-one mapping seems unavoidable and hard to tackle in the garden-hose model in general. Hence  $\text{ISA}_n$  appears to have an  $\Omega(n^2)$  garden-hose complexity.

<sup>7</sup> Some of the inequalities are straightforward but we explicitly state it for completeness.

<sup>8</sup> These exponential separations are in the standard communication model where the communication complexity is the *total* number of bits or qubits exchanged between Alice and Bob.

<sup>9</sup> We note that [15] defined a quantum version of the garden-hose model which differs from the classical model only by the ability of the players to have pre-shared entanglements. They used it to exhibit an exponential classical-quantum separation. Our definition of the quantum version of the memoryless model is a more natural generalization which involves exchanging quantum registers. Thus their exponential separation does not imply a similar separation in our model.

between QNM and NM using the standard idea of super-dense coding (which in fact applies to *all* classical memoryless protocols). Observe that by our garden-hose characterization earlier, this factor-2 separation translates to a quadratic separation between “quantum-garden-hose” model and the classical garden-hose model.

Since  $\text{NM}(F)$  is at most  $M^\rightarrow(F)$  for any  $F$ , memory-no memory communication complexity can be used to design garden-hose protocols. Using this method we obtain a sub-quadratic garden-hose protocol for computing the function *Disjointness with quadratic universe* which was conjectured to have a quadratic complexity in [25]. We discuss this protocol in Section 5.1.

**4. Towards obtaining better formula bounds.** Finally, it was shown by Klauck and Podder [25] that any formulae of size  $s$  consisting of *arbitrary* fan-in 2 gates (i.e., formulae over the binary basis of fan-in 2 gates) can be simulated by a garden-hose protocol of size  $s^{1+\varepsilon}$  for any arbitrary  $\varepsilon > 0$ . In this work, we show that an arbitrary garden-hose protocol can be simulated by a memoryless protocol without *any* additional loss, i.e., a size  $s$  garden-hose protocol can be turned into a memoryless protocol of size  $\log s$ . In particular, putting together these two connections, it implies that a size  $s$  formula can be turned into a memoryless protocol of size  $(1 + \varepsilon) \log s$ . Thus our result provides a new way of proving formulae size lower bound for arbitrary function  $F$  by analyzing the memoryless protocol of  $F$ .<sup>10</sup> The best known lower bound for formulae size (over the basis of all fan-in 2 gate) is  $\Omega(n^2/\log n)$ , due to Nečiporuk from 1966 [33]. Analogous to the Karchmer-Wigderson games [24] and Goldman and Håstad [22] techniques which uses communication complexity framework to prove circuit lower bounds our new communication complexity framework is a new tool for proving formulae size lower bounds. We note that in the memoryless model, constants really matter, e.g., a lower bound of  $\log n$  is not same as a lower bound of  $2 \log n$  as the former would give an  $n$  lower bound, whereas the latter will yield an  $n^2$  lower bound for the formula size. This is similar, in flavour, to the circuit depth lower bound where it took several decades of research to get from the trivial  $\log n$  lower bound to the sophisticated  $3 \log n$  lower bound by Håstad [22]. In formula size terminology this translates to going from  $n$  to  $n^3$ .

Brody et al. [11] conjectured that the problem of reachability in a graph requires  $\Omega(\log^2 n)$  non-oblivious memory. However as we have mentioned earlier the space-bounded communication complexity and the memoryless communication complexity of any function are equal up to a constant factor. Thus proving this conjecture would imply the same lower bound on the memoryless communication complexity and in turn imply an  $n^{\log n}$  formula-size lower bound for reachability, which would be a break-through in complexity theory. In fact, because of the same general formula to memoryless communication simulation, showing even a  $(2 + \varepsilon) \log n$  lower bound for reachability would be very interesting.

Finally an additional benefit to our characterization is the following: information theory has been used extensively to understand communication complexity [5, 6, 19, 10, 9] (just to cite a few references). As far as we are aware, usage of information theoretic techniques haven’t been explored when understanding the models of computation such as formula size, branching programs and garden-hose model. We believe our characterization using memoryless communication model might be an avenue to use information-theoretic ideas to prove stronger lower bounds in these areas.

<sup>10</sup> Here, the inputs  $x, y$  are distributed among two players and their goal is to compute  $(F \circ g)(x, y)$  where  $g$  is a constant-sized gadget.

**Open questions.** In this work, our main contribution has been to describe a seemingly-simple model of communication complexity and characterize its complexity using branching programs. We believe that our work could open up a new direction of research and results in this direction. Towards this, here we mention the natural open question (referring the reader to the full version for more open questions): Is there a function  $F : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$  and a universal constant  $c > 1$  for which we have  $\text{NM}(F) \geq c \log n$ . In particular, there are two consequences of such a result: (a) Using our relation to garden-hose model in Section 5.1, such a function will lead to the first *super-linear*  $n^c$  lower bound for garden-hose complexity, (b) using our characterized to branching programs, this would result in the first super-linear  $n^c$  lower bound for *bipartite* branching programs (analogous to Tal’s first super-linear lower bound on bipartite formula size<sup>11</sup> of inner-product [40]). Also if we could show this for  $c \geq 2 + \epsilon$ , this would imply a  $\Omega(n^{2+\epsilon})$  lower bound for general formula size, improving upon the best lower bound by Nečiporuk [33]. One possible candidate function which we haven’t been to rule out is the distributed 3-clique function: suppose Alice is given  $x \in \{0, 1\}^{\binom{n}{2}}$  and Bob is given  $y \in \{0, 1\}^{\binom{n}{2}}$ . We view their inputs as jointly labelling of the  $\binom{n}{2}$  edges of a graph on  $n$  vertices, then does the graph with edges labelled by  $x \oplus y$  have a triangle? Also, what is the complexity of the  $k$ -clique problem?

## 2 Preliminaries

**Notation.** Let  $[n] = \{1, \dots, n\}$ . For  $x \in \{0, 1\}^n$ , let  $\text{Int}(x) \in \{0, \dots, 2^n - 1\}$  be the integer representation of the  $n$ -bit string  $x$ . We now define a few standard functions which we use often in this paper. The equality function  $\text{EQ}_n : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$  is defined as  $\text{EQ}_n(x, y) = 1$  if and only if  $x = y$ . The disjointness function  $\text{DISJ}_n$  defined as  $\text{DISJ}_n(x, y) = 0$  if and only if there exists  $i$  such that  $x_i = y_i = 1$ . The inner product function  $\text{IP}_n$  is defined as  $\text{IP}(x, y) = \sum_i x_i \cdot y_i \pmod{2}$  (where  $\cdot$  is the standard bit-wise product).

We now define formulae, branching programs and refer the interested reader to Wegener’s book [41] for more on the subject.

► **Definition 2 (Branching programs (BP)).** A branching program for computing a Boolean function  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  is a directed acyclic graph with a source node labelled  $S$  and two sink nodes labelled 0 and 1. Every node except the source and sink nodes are labelled by an input variable  $x_i$ . The out-degree of every node is two and the edges are labelled by 0 and 1. The source node has in-degree 0 and the sink nodes have out-degree 0. The size of a branching program is the number of nodes in it. We say a branching program computes  $f$  if for all  $x \in f^{-1}(1)$  (resp.  $x \in f^{-1}(0)$ ) the algorithm starts from the source, and depending on the value of  $x_i \in \{0, 1\}$  at each node the algorithm either moves left or right and eventually reaches the 1-sink (resp. 0-sink) node. We denote  $\text{BP}(f)$  as the size (i.e., the number of nodes) of the smallest branching program that computes  $f$  for all  $x \in \{0, 1\}^n$ .

<sup>11</sup> Note that no super-linear lower bound is known for bipartite formulas that use all gates with fan-in 2 (in particular XOR gates).

### 3 Memoryless Communication Complexity

In this section we define memoryless communication complexity model and its variants.

#### 3.1 Deterministic Memoryless Communication Model

The crucial difference between the memoryless communication model and standard communication model is that, at any round of the communication protocol Alice and Bob do not have memory to remember previous transcripts and their private computations from the previous rounds. We now make this formal.

► **Definition 3** (Two-way Deterministic memoryless communication complexity). *Let  $F : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$ . Here there are two parties Alice and Bob whose goal is to compute  $F$ . Every  $s$ -bit memoryless protocol is defined by a set of functions  $\{f_x\}_{x \in \{0, 1\}^n}$  and  $\{g_y\}_{y \in \{0, 1\}^n}$  wherein  $f_x, g_y : \{0, 1\}^s \rightarrow \{0, 1\}^s$ . On input  $x, y$  to Alice and Bob respectively a memoryless protocol is defined as follows: at every round Alice obtains a message  $m_B \in \{0, 1\}^s$  from Bob, she computes  $m_A = f_x(m_B) \in \{0, 1\}^s$  and sends  $m_A$  to Bob. On receiving  $m_A$ , Bob computes  $m'_B = g_y(m_A)$  and replies with  $m'_B \in \{0, 1\}^s$  to Alice. They alternately continue doing this for every round until the protocol ends. Without loss of generality we assume the protocol ends once  $m_A, m_B \in \{1^{s-1}0, 1^{s-1}1\}$ , then the function output is given by the last bit. So, once the transcript is  $1^{s-1}b$ , Alice and Bob output  $F(x, y) = b$ .<sup>12</sup>*

*We say a protocol  $P_F$  computes  $F$  correctly if for every  $(x, y)$ , Bob outputs  $F(x, y)$ . We let  $\text{cost}(P_F, x, y)$  be the smallest  $s$  for which  $P_F$  computes  $F$  on input  $(x, y)$ . Additionally, we let*

$$\text{cost}(P_F) = \max_{x, y} \text{cost}(P_F, x, y)$$

*and the memoryless communication complexity of computing  $F$  in this model is defined as*

$$\text{NM}(F) = \min_{P_F} \text{cost}(P_F),$$

*where the minimum is taken over all protocols  $P_F$  that compute  $F$  correctly.*

We crucially remark that in the memoryless model, the players do not even have access to a clock and hence they cannot tell which round of the protocol they are in. At every round they just compute their local functions  $\{f_x\}_x, \{g_y\}_y$  on the message they received and proceed according to the output of these functions.

**One-way Deterministic Memoryless Model.** Similar to the definition above, one can define the *one-way* memoryless communication complexity wherein only Alice is allowed to send messages to Bob and the remaining aspects of this model is the same as Definition 3. We denote the complexity in this model by  $\text{NM}^\rightarrow(F)$ . It is easy to see that since Alice does not have any memory she cannot send multi-round messages to Bob as there is no way for her to remember in which round she is in. Also Bob cannot send messages back to Alice for her to keep a clock. Hence all the information from Alice to Bob has to be conveyed in a single round. Thus one-way memoryless communication complexity is equal to the standard deterministic one-way communication complexity.<sup>13</sup>

► **Fact 4.** *For all function  $F$  we have  $\text{NM}^\rightarrow(F) = D^\rightarrow(F)$ .*

<sup>12</sup> Without loss of generality, we assume that the first message is between Alice and Bob and she sends  $f_x(0^s) \in \{0, 1\}^s$  to Bob.

<sup>13</sup> Without loss of generality, in any one-way standard communication complexity protocol of cost  $c$  Alice can send all the  $c$  bits in a single round.

### 3.2 Deterministic Memory-No Memory Communication Model

We now consider another variant of the memoryless communication model wherein one party is allowed to have a memory but the other party doesn't. In this paper, we always assume that Alice has a memory and call this setup the *memory no-memory model*. In this work, we will *not* consider the other case wherein Bob has a memory and Alice doesn't have a memory. Note that this setting is asymmetric i.e., there exists functions for which the complexity of the function can differ based on whether Alice or Bob has the memory.

**Two-way Memory-No Memory Communication Model.** Here the players are allowed to send messages in both directions. For a function  $F : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$ , we denote the complexity in this model as  $M(F)$ . Observe that  $M(F)$  is trivially upper bounded by  $\log n$  for every  $F$ : for every  $i \in [n]$ , Alice can send  $i$  and Bob replies with  $y_i$ . Since Alice has memory, after  $n$  rounds she has complete knowledge of  $y \in \{0, 1\}^n$  and computes  $F(x, y)$  locally and sends it to Bob.

**One-way Memory-No Memory Communication Model.** Here we allow only Alice to send messages to Bob. Since Alice has a memory she can send multiple messages one after another, but Bob cannot reply to her messages. Hence, after receiving any message Bob computes the function  $g_y(\cdot) \in \{0, 1, \perp\}$  and if he obtains  $\{0, 1\}$ , he outputs 0 or 1, and continues if he obtains  $\perp$ . We denote the communication complexity in this model by  $M^\rightarrow(F)$ . This model was formally studied by Papakonstantinou et al. [35] as *overlay communication complexity* (we discuss their main contributions in Section 4).

Finally, we can also have a model where both players have memory and hence both players can remember the whole transcript of the computation. This is exactly the widely-studied standard communication complexity except that the complexity measure here is the size of the *largest* transcript (so the complexity in our model is just 1 since they could exchange a single bit for  $n$  rounds and compute an arbitrary function on  $2n$  bits) and the latter counts the *total* number of bits exchanged in a protocol.

**Quantum memoryless Models.** Here we introduce the quantum memoryless communication model. There are a few ways one can define the quantum extension of the classical memoryless model. We find the following exposition the simplest to explain. This quantum communication model is defined exactly as the classical memoryless model except that Alice and Bob are allowed to communicate *quantum* states. A  $T$  round quantum protocol consists of the following: Alice and Bob have local  $k$ -qubit memories  $A, B$  respectively,<sup>14</sup> they share a  $m$ -qubit message register  $M$  and for every round they perform a  $q$ -outcome POVM  $\mathcal{P} = \{P_1, \dots, P_q\}$  for  $q = 2^m$  (which could potentially depend on their respective inputs  $x$  and  $y$ ). Let  $\{U_x\}_{x \in \{0, 1\}^n}, \{V_y\}_{y \in \{0, 1\}^n}$  be the set of  $(m + k)$ -dimensional unitaries acting on  $(A, M)$  and  $(B, M)$  respectively (this is analogous to the look-up tables  $\{f_x, g_y : \{0, 1\}^m \rightarrow \{0, 1\}^m\}_{x, y \in \{0, 1\}^n}$  used by Alice and Bob in the classical memoryless protocol). Let  $\psi_0 = (A, M)$  be the all-0 mixed state. Then, the quantum protocol between Alice and Bob can be written as follows: on input  $x, y$  to Alice and Bob respectively, on the  $i$ th round (for  $i \geq 1$ ) Alice sends  $\psi_i$  for odd  $i$  and Bob replies with  $\psi_{i+1}$  defined as follows:

$$\psi_i = \text{Tr}_A(\mathcal{P} \circ U_x \psi_{i-1}) \otimes |0\rangle\langle 0|_B,$$

where  $\mathcal{P} \circ U_x \psi_{i-1}$  is the post-measurement state after performing the POVM  $\mathcal{P}$  on the state  $U_x \psi_{i-1}$  and  $\text{Tr}_A(\cdot)$  refers to taking the partial trace of register  $A$ . Similarly, define

<sup>14</sup> After each round of communication, these registers are set to the all-0 register.

$$\psi_i = |0\rangle\langle 0|_A \otimes \text{Tr}_B(\mathcal{P} \circ U_y \psi_i),$$

where  $\text{Tr}_B(\cdot)$  takes the partial trace of register B. Intuitively, the states  $\psi_i$  (similarly  $\psi_{i+1}$ ) can be thought of as follows: after applying unitaries  $U_x$  to the registers (A, M), Alice applies the  $q$ -outcome POVM  $\mathcal{P}$  which results in a classical outcome and post-measurement state on the registers (A, M) and she discards her private memory register and initializes the register B in the all-0 state. The quantum communication protocol terminates at the  $i$ th round once the  $q$ -outcome POVM  $\mathcal{P}$  results in the classical outcome  $\{(1^{m-1}, b)\}_{b \in \{0,1\}}$ .<sup>15</sup> After they obtain this classical output, Alice and Bob output  $b$ . We say a *protocol computes  $F$*  if for every  $x, y \in \{0,1\}^n$ , with probability at least  $2/3$  (probability taken over the randomness in the protocol), after a certain number of rounds the POVM measurement results in  $(1^{m-1}, F(x, y))$ . The complexity of computing  $F$  in the quantum memoryless model, denoted  $\text{QNM}(F)$  is the smallest  $m$  such that there is a  $m$ -qubit message protocol that computes  $F$ . As defined before, we also let  $\text{QM}^\rightarrow(F)$  (resp.  $\text{QNM}^\rightarrow(F)$ ) to be the model in which Alice has a memory (has no memory) and Bob doesn't have a memory and the communication happens from Alice to Bob.

Note that unlike the classical case, with quantum messages there is no apparent way for the players to know with certainty if they have received a designated terminal state (and whether they should stop and output 0/1) without disturbing the message content. Thus a natural choice is to integrate a partial measurement of the message register at each round into the definition.

**Notation.** For the remaining part of the paper we abuse notation by letting  $\text{NM}(F)$ ,  $\text{QNM}(F)$  denote the memoryless *complexity* of computing  $F$  and we let **NM model** (resp. **QNM model**) be the memoryless communication model (resp. quantum memoryless communication model). Additionally, we omit *explicitly* writing that Alice and Bob exchange the final message  $1^{s-1}f(x, y)$  once either parties have computed  $f(x, y)$  (on input  $x, y$  respectively).

## 4 Understanding and characterization of memoryless models

We now state a few observations and relations regarding the memoryless communication models.

► **Fact 5.** *For every  $F : \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}$ , we have  $\text{M}(F) \leq \text{NM}(F) \leq 2\text{M}^\rightarrow(F) \leq 2\text{NM}^\rightarrow(F)$ .*

We refer the reader to the full version of the paper for the proof. As we mentioned earlier, our main contribution in this paper is the memoryless **NM** model of communication. We saw in Fact 4 that  $\text{NM}^\rightarrow(F)$  is equal to the standard one-way deterministic communication complexity of computing  $F$ . The  $\text{M}^\rightarrow(F)$  model was introduced and studied by Papakonstantinou et al. [35]. Additionally observe that the strongest model of communication complexity  $\text{M}(F)$  is small for every function  $F$ .

► **Fact 6.** *For every  $F : \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}$ , we have  $\text{M}(F) \leq \log n$ .*

<sup>15</sup> We remark that a good quantum communication protocol should be such that for every  $i \in [T]$ , the probability of obtaining  $(1^{m-1}, 1 \oplus F(x, y))$  when measuring  $\psi_i$  using the POVM  $\mathcal{P}$  should be  $\leq 1/3$ .



To see this, observe that in the M model (i.e., two-way memory-no memory model), on the  $i$ th round, Alice sends  $i \in [n]$  and Bob (who doesn't have memory) sends the message  $y_i$  to Alice. Alice stores  $y_i$  and increments  $i$  to  $i + 1$  and repeats. After  $n$  rounds Alice simply has the entire  $y$  and computes  $F(x, y)$  on her own (note that  $F$  is known to both Alice and Bob).

Below we give few protocols in the NM model to give more intuition of this model.

**Algorithms in the memoryless model.** In the introduction we described a  $\log n + 1$  protocol for the equality function. Below we describe a protocol for the inner product function. For the inner product function  $\text{IP}_n$ , a simple protocol is as follows: For  $i = 1, \dots, n$ , on the  $i$ th round, Alice sends  $(i, x_i, \sum_{j=0}^{i-1} x_i \cdot y_j \pmod{2})$  which takes  $\log n + 2$  bits and Bob replies with  $(i, x_i, \sum_{j=0}^{i-1} x_i \cdot y_j + x_i \cdot y_i \pmod{2}) = (i, x_i, \sum_{j=0}^i x_i \cdot y_j \pmod{2})$ .<sup>16</sup> They repeat this protocol for  $n$  rounds and after the  $n$ th round, they have computed  $\text{IP}_n(x, y)$ . Hence  $\text{NM}(\text{IP}_n) \leq \log n + 2$ . Now we describe a protocol for the disjointness function  $\text{DISJ}_n$ . Here a  $\log n$  protocol is as follows: Alice sends the first coordinate  $i \in [n]$  for which  $x_i = 1$  and Bob outputs 0 if  $y_i = 1$ , if not Bob replies with the first  $j$  after  $i$  for which  $y_j = 1$  and they repeat this procedure until  $i$  or  $j$  equals  $n$ . It is not hard to see that  $\text{DISJ}_n(x, y) = 0$  if and only if there exists  $k$  for which  $x_k = y_k = 1$  in which case Alice and Bob will find such (smallest)  $k$  in the protocol above, if not the protocol will run for at most  $n$  rounds and they decide that  $\text{DISJ}_n(x, y) = 1$ . We now mention a non-trivial protocol in the NM model for the majority function defined as  $\text{MAJ}_n(x, y) = [\sum_i x_i \cdot y_i \geq n/2 + 1]$ . A trivial protocol for  $\text{MAJ}_n$  is similar to the  $\text{IP}_n$  protocol, on the  $(i + 1)$ th round, Alice sends  $(i, x_i, \sum_{i=1}^n x_i y_i)$  (without the  $\pmod{2}$ ) and Bob replies with  $(i, x_i, \sum_{i=1}^{n+1} x_i \cdot y_i)$ . Note that this protocol takes  $2 \log n + 1$  bits ( $\log n$  for sending the index  $i \in [n]$  and  $\log n$  to store  $\sum_{i=1}^n x_i \cdot y_i \in [n]$ ). Apriori this seems the best one can do, but interestingly using intricate ideas from number theory there exists a  $n \log^3 n$  [37, 25] garden-hose protocol for computing  $\text{MAJ}_n$ . Plugging this in with Theorem 21 we get a protocol of cost  $\log n + 3 \log \log n$  for computing  $\text{MAJ}_n$  in the NM model.

An interesting question is, are these protocols for  $\text{IP}_n, \text{EQ}_n, \text{DISJ}_n, \text{MAJ}_n$  optimal? Are there more efficient protocols possibly with constant bits of communication in each round? In order to understand this, in the next section we show that the memoryless communication complexity is lower bounded by the standard deterministic one-way communication complexity. Using this connection, we can show the tightness of the first three protocols. Additionally, we show that  $\text{NM}(\text{MAJ}_n) \geq \log n$ , thus the exact status of  $\text{NM}(\text{MAJ}_n) \in \{\log n, \dots, \log n + 3 \log \log n\}$  remains an intriguing open question.

## 4.1 Lower bounds on memoryless communication complexity

In the introduction, we mentioned that it is an interesting open question to find an explicit function  $F$  for which  $\text{NM}(F) \geq 2 \log n$ . Unfortunately we do not even know of an explicit function for which we can prove lower bounds better than  $\log n + \omega(1)$  (we discuss more about this in the open questions). However, it is not hard to show that for a random function  $F$ , the memoryless communication complexity of  $F$  is large.

► **Lemma 7.** *Let  $F : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$  be a random function. Then,  $\text{NM}(F) = \Omega(n)$ .*

<sup>16</sup>Technically Bob need not send back the bit  $x_i$ .



The proof is via a simple counting argument. We refer the reader to the full version of the paper for the definition. We remark that similar ideas used in this lemma can be used to show that for all  $s < s'$ , there exists functions that can be computed using  $s'$  bits of communication in each round but not  $s$  bits of communication. This gives rise to a *space hierarchy* theorem for the NM model.

#### 4.1.1 Deterministic one-way communication complexity and memoryless complexity

We now give a very simple lower bound technique for the memoryless communication model in terms of deterministic one-way communication. Although this lower bound is “almost immediate”, as we mentioned in the introduction, it already gives us non-trivial lower bounds on the NM complexity of certain functions.

► **Fact 8.**

$$\text{NM}(F) \geq \log \left( D^{\rightarrow}(F) / \log D^{\rightarrow}(F) \right), \text{ and } \text{QNM}(F) \geq \Omega \left( \log \left( D^{\rightarrow}(F) / \log D^{\rightarrow}(F) \right) \right).$$

Using this lemma, we immediately get the following corollary.

► **Corollary 9.** *Let  $n \geq 2$ . Then  $\text{NM}(\text{EQ}_n), \text{NM}(\text{IP}_n), \text{NM}(\text{DISJ}_n), \text{NM}(\text{MAJ}_n), \text{NM}(\text{Index}), \text{NM}(\text{BHM})$  is  $\Omega(\log n)$ . Similarly, we have QNM complexity of these functions are  $\Omega(\log n)$ .*

This corollary follows immediately from Fact 8 because the deterministic-one way communication complexity of these functions are at least  $n$  (by a simple adversarial argument), thereby showing that the  $(\log n)$ -bit protocols we described in the beginning of this section for the first three of these functions is close-to-optimal. However one drawback of Fact 8 is it cannot be used to prove a lower bound that is better than  $\log n$  since  $D^{\rightarrow}(F) \leq n$  for every function  $F : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$ .

## 4.2 Characterization of memoryless communication

Papakonstantinou et al. [35] consider the memory-nomemory model of communication complexity wherein Alice has a memory and Bob doesn't and they are restricted to one-way communication from Alice to Bob. They show a beautiful combinatorial rectangle-overlay characterization (denoted  $\text{RO}(F)$ ) of the  $M^{\rightarrow}$  model. We refer the reader to the full version of the paper for the definition.

One of the main results of [35] was the following characterization.

► **Theorem 10 ([35]).** *For every  $F$ , we have  $\log \text{RO}(F) \leq M^{\rightarrow}(F) \leq 2 \log \text{RO}(F)$ .*

A natural question following their work is, can we even characterize our new general framework of communication complexity wherein *both* Alice and Bob do not have memory and the communication can be two-way. Generalizing the rectangle-based characterization of [35] to our setting seemed non-trivial because in our communication model the memoryless-ness of the protocol doesn't seem to provide any meaningful way to split the communication matrix into partitions or overlays (as far as we could analyze). Instead we characterize our communication model in terms of *bipartite branching programs*, which we define below.<sup>17</sup>

---

<sup>17</sup> For a definition of general branching program (BP), refer to Section 2.

► **Definition 11** (Bipartite Branching Program (BBP)). *Let  $F : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$ . A bipartite branching program is a BP that computes  $F$  in the following way: for every  $(x, y)$ , each node in the branching program is either labelled by a function  $f_i \in \mathcal{F} = \{f_i : \{0, 1\}^n \rightarrow \{0, 1\}\}_i$  or by  $g_j \in \mathcal{G} = \{g_j : \{0, 1\}^n \rightarrow \{0, 1\}\}_j$ ; the output edge is labelled by 0 or 1 and the output of the function in the node label decides which edge to follow. The size of a BBP is the number of nodes in it. We define  $\text{BBP}(F)$  as the size of the smallest program that computes  $F$  for all  $(x, y) \in \{0, 1\}^{2n}$ .*

Observe that in a BBP every node no longer just queries  $x \in \{0, 1\}^n$  at an arbitrary index  $i$  (like in the standard BP), but instead is allowed to compute an *arbitrary Boolean function* on  $x$  or  $y$ . Of course, another natural generalization of BBP is, why should the nodes of the program just compute Boolean-valued functions? We now define the *generalized BBP* wherein each node can have out-degree  $k$  (instead of out-degree 2 in the case of BBP and BP).

► **Definition 12** (Generalized Bipartite Branching Program (GBBP)). *Let  $k \geq 1$ . A generalized bipartite branching program is a BBP that computes  $F$  in the following way: for every  $(x, y)$ , each node in the branching program can have out-degree  $k$  and labelled by the node  $f_i \in \mathcal{F} = \{f_i : \{0, 1\}^n \rightarrow [k]\}_i$ , or by  $g_j \in \mathcal{G} = \{g_j : \{0, 1\}^n \rightarrow [k]\}_j$ ; the output edges are labelled by  $\{1, \dots, k\}$  and the output of the function in the node label decides which edge to follow. The size of a GBBP is the number of nodes in it. We define  $\text{GBBP}(F)$  as the size of the smallest program that computes  $F$  for all  $(x, y) \in \{0, 1\}^{2n}$ .*

We now show that the generalized bipartite branching programs are not much more powerful than bipartite branching programs, in fact these complexity measures are quadratically related.

► **Fact 13.** *For  $F : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$ , we have  $\text{GBBP}(F) \leq \text{BBP}(F) \leq \text{GBBP}(F)^2$ .*

It is not clear if the quadratic factor loss in the simulation above is necessary and we leave it as an open question. We are now ready to prove our main theorem relating NM communication model and bipartite branching programs.

► **Theorem 14.** *For every  $F$ , we have  $\frac{1}{2} \log \text{BBP}(F) \leq \text{NM}(F) \leq \log \text{BBP}(F)$ .*

We refer the reader to the full version of the paper for the proof. Earlier we saw that GBBP is polynomially related to BBP. We now observe that both these measures can be exponentially smaller than standard branching program size.<sup>18</sup>

► **Fact 15.** *The parity function  $\text{PARITY}_n(x, y) = \sum_i x_i \oplus y_i \pmod{2}$  gives an exponential separation between generalized bipartite branching programs and branching programs.*

**Time Space Trade-off for Memoryless.** Finally, we mention a connection between our communication model and time-space trade-offs. In particular, what are the functions that can be computed if we limit the number of rounds in the memoryless protocol? Earlier we saw that, an arbitrary memoryless protocol of cost  $s$  for computing a function  $F$  could consist of at most  $2^{s+1}$  rounds of message exchanges. If sending one message takes one unit of time, we can ask whether it is possible to simultaneously reduce the message size  $s$  and the time  $t$  required to compute a function. The fact below gives a time-space trade-off in terms of deterministic communication complexity.

<sup>18</sup>The function we use here is the standard function that separates bipartite formula size from formula size.

► **Fact 16.** *For every  $k \geq 1$  and function  $F : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$ , we have  $\text{NM}_k(F) \geq D(F)/k$ , where  $\text{NM}_k(F)$  is the NM communication complexity of computing  $F$  with at most  $k$  rounds of communication, and  $D(F)$  is the standard deterministic communication complexity.*

It is not hard to now see that the number of rounds in an  $\text{NM}(F)$  protocol corresponds to the *depth* of the generalized bipartite branching program computing  $F$ . So an immediate corollary of the fact above is, even for simple functions such as equality, inner product, if we restrict the depth of GBBP to be  $o(n)$ , then we can show *exponential-size* lower bounds on such GBBPs computing these functions. Similarly note that one can separate QNM and NM model of communication if we bound the number of rounds: consider the problem where Alice and Bob get  $x, y \in \{0, 1\}^n$  respectively promised that,  $x = y$  or Hamming distance between  $x, y$  is  $n/2$ . In this case, clearly  $\text{NM}_k(F) \geq n/k$  (from the fact above), which in particular means that constant-round NM protocols need to send  $\Omega(n)$  bits. In contrast, in the QNM model, Alice could simply send  $O(1)$  copies of a fingerprint state  $|\psi_x\rangle = \frac{1}{\sqrt{n}} \sum_i (-1)^{x_i} |i\rangle$  (in a *single round*) and due to the promise, Bob can perform swap test between  $|\psi_x\rangle, |\psi_y\rangle$  and decide if  $x = y$  or the Hamming distance is  $n/2$  with probability 1.

## 5 Relations between memoryless communication models

In this section, we show that there exists exponential separations between the four memoryless communication models defined in Section 3 (and in particular, Fact 5).

► **Theorem 17.** *There exists functions  $F$  for which the following inequalities (as shown in Fact 5) is exponentially weak<sup>19</sup>  $M(F) \leq \text{NM}(F) \leq 2M^\rightarrow(F) \leq 2\text{NM}^\rightarrow(F)$ .*

We refer the reader to the full version of the paper for the definition. We now exhibit exponential separations between the quantum and classical memoryless models of communication complexity.

► **Theorem 18.** *There exist functions  $F : D \rightarrow \{0, 1\}$  where  $D \subseteq \{0, 1\}^n \times \{0, 1\}^n$  for which the following inequalities are exponentially weak: (i)  $\text{QNM}^\rightarrow(F) \leq \text{NM}^\rightarrow(F)$ , (ii)  $\text{QM}^\rightarrow(F) \leq M^\rightarrow(F)$ , (iii)  $\text{QM}(F) \leq M(F)$ .<sup>20</sup>*

We refer the reader to the full version of the paper for the definition. One drawback in the exponential separations above is that we allow a quantum protocol to err with constant probability but require the classical protocols to be correct with probability 1. We remark that except the second inequality, the remaining inequalities also show exponential separations between the randomized memoryless model (wherein Alice and Bob have public randomness and are allowed to err in computing the function) versus the corresponding quantum memoryless model. A natural question is to extend these separations even when the classical model is allowed to err with probability at least  $1/3$ .

### 5.1 Relating the Garden-hose model, Space-bounded communication complexity and Memoryless complexity

In this section, we show that the memoryless communication complexity  $\text{NM}(F)$  of a Boolean function  $F$  is equal to the logarithm of the garden-hose complexity up to an additive constant and is equal to the space-bounded communication complexity up to factor 2. But first we briefly define the garden-hose model and the space-bounded communication complexity model.

<sup>19</sup>We remark that the functions exhibiting these exponential separations are different for the three inequalities.

<sup>20</sup>Again, the functions exhibiting these separations are different for the three inequalities.

**Garden-hose model [15].** In the garden-hose model of computation, Alice and Bob are neighbours (who cannot communicate) and have few pipes going across the boundary of their gardens. Based on their private inputs  $x, y$  and a function  $F : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$  known to both, the players connect some of the opening of the pipes on their respective sides with garden-hoses. Additionally, Alice connects a tap to one of the pipes on her side. Naturally, based on the garden-hose connections, water travels back and forth through some of the pipes and finally spills on either Alice's or Bob's side, based on which they decide if a function  $F$  on input  $x, y$  evaluates to 0 or 1. It is easy to show that Alice and Bob can compute every function using this game. The garden-hose complexity  $\text{GH}(F)$  is defined to be the minimum *number of pipes* required to compute  $F$  this way for all possible inputs  $x, y$  to Alice and Bob. For more on garden-hose complexity, we refer the interested reader to [15, 25, 38, 39].

**Space-bounded communication complexity [11].** Alice and Bob each have at most  $s(n)$  bits of memory. Based on their private inputs  $x, y$  they want to compute the function  $F : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$  in the following manner: At each round Alice receives a single bit message  $m_B \in \{0, 1\}$  from Bob and based on her input  $x$ , the incoming message  $m_B$  and her previous  $s(n)$ -bit register content, she computes a new  $s(n)$ -bit register and decides whether to stop and output 0/1 or to continue. Bob does the same. At the beginning of the game, the register contents of both players are set to the all-zero strings. The game then starts by Alice sending the first message and continues until one of players outputs 0/1. space-bounded communication complexity  $\text{SM}(F)$  of computing a function  $F$  is the minimum register size  $s(n)$  required to compute  $F$  on the worst possible input  $(x, y)$ . Brody et al. [11] claimed that space-bounded communication complexity is equal to the garden-hose communication complexity upto factor 2.

▷ **Claim 19 ([11]).** For every function  $F$  there exists constants  $c \in (0, 1), d \in \mathbb{N}^+$  such that  $c \cdot 2^{\text{SM}(F)} \leq \text{GH}(F) \leq 2^{2\text{SM}(F)+2} + d$ .

We show the following relation between

► **Lemma 20.** For every function  $F$ ,  $\text{NM}(F) \leq 2\text{SM}(F) + 1$ ,  $\text{SM}(F) \leq \text{NM}(F) + \log \text{NM}(F)$

We refer the reader to the full version of the paper for a proof. Using the Claim 19 and Lemma 20 we can conclude that the logarithm of the garden-hose complexity is equal to the memoryless NM complexity up to factor 2. This seems interesting already given we can connect these two models, but in the NM model, even factor-2s are important since they are related to formula lower bounds. Now we show that it is possible to further tighten the relation in the lemma above. Below we show that NM is actually equivalent to the logarithm of the garden-hose complexity up to an *additive term* of 4. The first observation relating the garden-hose model and memoryless communication complexity is that, the garden-hose model is exactly the NM communication model, except that in addition to the memoryless-ness of Alice and Bob, there is a *bijection* between the incoming and the outgoing messages of both players (i.e., the local functions Alice and Bob apply  $\{f_x : \{0, 1\}^s \rightarrow \{0, 1\}^s\}_x, \{g_y : \{0, 1\}^s \rightarrow \{0, 1\}^s\}_y$  are *bijective functions*). We now state and prove the theorem which shows how GH is related to the standard memoryless communication model.

► **Theorem 21.** For  $F$ , we have  $\log \text{GH}(F) - 4 \leq \text{NM}(F) \leq \log \text{GH}(F)$ .

We refer the reader to the full version of the paper for a proof. Interestingly, Theorem 21 together with Theorem 17 gives us a way to construct a garden-hose protocol using an  $\text{M}^\rightarrow$  protocol and, as we will see below, this could result in potentially stronger upper bound on

the garden-hose model. In an earlier work of Klauck and Podder [25], it was conjectured that the disjointness function with input size  $m = n \cdot 2 \log n$  (i.e., with set size  $n$  and universe size  $n^2$ ) has a quadratic lower bound  $\Omega(m^2)$  in the garden-hose model. Here, we show that GH protocol for this problem has cost  $O(m^2/\log^2 m)$ . Although the improvement is only by a logarithmic-factor, we believe that this complexity can be reduced further which we leave as an open question.

**Disjointness with quadratic universe:** Alice and Bob are given  $n$  numbers each from  $[n^2]$  as a  $m = n \cdot 2 \log n$  long bit strings. Their goal is to check if all of their  $2n$  numbers are unique. Without loss we can assume that the  $n$  numbers on the respective sides of Alice and Bob are unique, if not they can check it locally and output 0 without any communication. Then an  $M^\rightarrow$  protocol for computing this function is as follows: Alice keeps sending all her numbers to Bob one by one (using her local memory to keep track of which numbers she has already sent). This requires  $2 \log n$  size message register on every round. Bob upon receiving any number from Alice, checks if any number of his side matches the number received. If there is a match he outputs 0, else he continues. For the last message Alice sends the number along with a special marker. Bob performs his usual check and output 1 if the check passes and the marker is present. Clearly the cost of this protocol is  $2 \log n$  and thus from Theorem 21 the garden-hose protocol for computing this function has cost  $n^2$ . Since the input size is  $m = n \cdot 2 \log n$ , the cost of the garden-hose protocol is  $O(m^2/\log^2 m)$ .

---

## References

- 1 N. Alon, Y. Matias, and M. Szegedy. The space complexity of approximating the frequency moments. *Journal on Computer Systems and Science*, 58(1):137–147, 1999.
- 2 S. Arunachalam and S. Podder. Communication memento: Memoryless communication complexity. *arXiv*, 2000. [arXiv:2005.04068](#).
- 3 L. Babai, P. Frankl, and J. Simon. Complexity classes in communication complexity theory. In *27th Annual Symposium on Foundations of Computer Science (sfcs 1986)*, pages 337–347. IEEE, 1986.
- 4 Z. Bar-Yossef, T. S. Jayram, and I. Kerenidis. Exponential separation of quantum and classical one-way communication complexity. *SIAM Journal on Computing*, 38(1):366–384, 2008. Earlier in STOC’04.
- 5 Ziv Bar-Yossef, Thathachar S Jayram, Ravi Kumar, and D Sivakumar. Information theory methods in communication complexity. In *Proceedings 17th IEEE Annual Conference on Computational Complexity*, pages 93–102. IEEE, 2002.
- 6 Ziv Bar-Yossef, Thathachar S Jayram, Ravi Kumar, and D Sivakumar. An information statistics approach to data stream and communication complexity. *Journal of Computer and System Sciences*, 68(4):702–732, 2004.
- 7 E. Blais, J. Brody, and K. Matulef. Property testing lower bounds via communication complexity. *Computational Complexity*, 21(2):311–358, 2012.
- 8 M. Braverman and O. Weinstein. A discrepancy lower bound for information complexity. *Algorithmica*, 76(3):846–864, 2016.
- 9 Mark Braverman. Interactive information complexity. *SIAM Journal on Computing*, 44(6):1698–1739, 2015.
- 10 Mark Braverman, Ankit Garg, Denis Pankratov, and Omri Weinstein. From information to exact communication. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 151–160, 2013.
- 11 J. E. Brody, S. Chen, P. A. Papakonstantinou, H. Song, and X. Sun. Space-bounded communication complexity. In *Proceedings of the 4th conference on Innovations in Theoretical Computer Science*, pages 159–172, 2013.

- 12 H. Buhrman, R. Cleve, J. Watrous, and R. de Wolf. Quantum fingerprinting. *Physical Review Letters*, 87(16):167902, 2001.
- 13 H. Buhrman, R. Cleve, and A. Wigderson. Quantum vs. classical communication and computation. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing (STOC)*, pages 63–68, 1998.
- 14 H. Buhrman, L. Czekaj, A. Grudka, M. Horodecki, P. Horodecki, M. Markiewicz, F. Speelman, and S. Strelchuk. Quantum communication complexity advantage implies violation of a bell inequality. *Proceedings of the National Academy of Sciences*, 113(12):3191–3196, 2016.
- 15 H. Buhrman, S. Fehr, C. Schaffner, and F. Speelman. The garden-hose model. In *Innovations in Theoretical Computer Science, ITCS*, pages 145–158, 2013.
- 16 Well Y Chiu, Mario Szegedy, Chengu Wang, and Yixin Xu. The garden hose complexity for the equality function. In *International Conference on Algorithmic Applications in Management*, pages 112–123. Springer, 2014.
- 17 Yfke Dulek, Christian Schaffner, and Florian Speelman. Quantum homomorphic encryption for polynomial-sized circuits. In *Annual International Cryptology Conference*, pages 3–32. Springer, 2016.
- 18 S. Fiorini, S. Massar, S. Pokutta, H. R. Tiwary, and R. de Wolf. Exponential lower bounds for polytopes in combinatorial optimization. *Journal of the ACM*, 62(2):17:1–17:23, 2015.
- 19 Anat Ganor, Gillat Kol, and Ran Raz. Exponential separation of information and communication for boolean functions. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 557–566, 2015.
- 20 D. Gavinsky. Bare quantum simultaneity versus classical interactivity in communication complexity. *arXiv preprint*, 2019. [arXiv:1911.01381](https://arxiv.org/abs/1911.01381).
- 21 D. Gavinsky, J. Kempe, I. Kerenidis, R. Raz, and R. de Wolf. Exponential separation for one-way quantum communication complexity, with applications to cryptography. *SIAM Journal on Computing*, 38(5):1695–1708, 2008. Previous in FOCS’07.
- 22 Mikael Goldmann and Johan Håstad. A simple lower bound for monotone clique using a communication game. *Information Processing Letters*, 41(4):221–226, 1992.
- 23 R. Impagliazzo and R. Williams. Communication complexity with synchronized clocks. In *2010 IEEE 25th Annual Conference on Computational Complexity*, pages 259–269. IEEE, 2010.
- 24 M. Karchmer and A. Wigderson. Monotone circuits for connectivity require super-logarithmic depth. *SIAM Journal on Discrete Mathematics*, 3(2):255–265, 1990.
- 25 H. Klauck and S. Podder. New bounds for the garden-hose model. In *34th International Conference on Foundation of Software Technology and Theoretical Computer Science, FSTTCS*, pages 481–492, 2014.
- 26 Hartmut Klauck. Quantum and classical communication-space tradeoffs from rectangle bounds. In *International Conference on Foundations of Software Technology and Theoretical Computer Science*, pages 384–395. Springer, 2004.
- 27 E. Kushilevitz and N. Nisan. *Communication complexity*. Cambridge University Press, 1997.
- 28 T. Lam, P. Tiwari, and M. Tompa. Tradeoffs between communication and space. In *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 217–226, 1989.
- 29 Tak Lam, Prasoon Tiwari, and Martin Tompa. Trade-offs between communication and space. *Journal of Computer and System Sciences*, 45(3):296–315, 1992.
- 30 Klaus-Jörn Lange, Pierre McKenzie, and Alain Tapp. Reversible space equals deterministic space. *Journal of Computer and System Sciences*, 60(2):354–367, 2000.
- 31 T. Lee and A. Shraibman. Lower bounds in communication complexity. *Foundations and Trends in Theoretical Computer Science*, 3(4):263–398, 2009.
- 32 P. B. Miltersen, N. Nisan, S. Safra, and A. Wigderson. On data structures and asymmetric communication complexity. *Journal on Computer Systems and Science*, 57(1):37–49, 1998.
- 33 Edward I Nechiporuk. A boolean function. *Engl. transl. in Sov. Phys. Dokl.*, 10:591–593, 1966.
- 34 M. Pal. Communication complexity and parallel computing. *IEEE Concurrency*, 7(2):81–83, 1999.

- 35    P. A. Papakonstantinou, D. Scheder, and H. Song. Overlays and limited memory communication. In *IEEE 29th Conference on Computational Complexity, CCC*, pages 298–308, 2014.
- 36    R. Paturi and J. Simon. Probabilistic communication complexity. *Journal of Computer and System Sciences*, 33(1):106–123, 1986.
- 37    R. K. Sinha and J. S. Thathachar. Efficient oblivious branching programs for threshold and mod functions. *Journal of Computer and System Sciences*, 55(3):373–384, 1997.
- 38    F. Speelman. Position-based quantum cryptography and the garden-hose game, 2012. Masters Thesis.
- 39    F. Speelman. Position-based quantum cryptography and catalytic computation, 2016. PhD Thesis.
- 40    A. Tal. The bipartite formula complexity of inner-product is quadratic, 2016. Earlier in STOC’17.
- 41    I. Wegener. *The complexity of Boolean functions*. BG Teubner, 1987.
- 42    R. de Wolf. Nondeterministic quantum query and communication complexities. *SIAM Journal on Computing*, 32(3):681–699, 2003.
- 43    A. C. Yao. Quantum circuit complexity. In *Proceedings of 1993 IEEE 34th Annual Foundations of Computer Science*, pages 352–361. IEEE, 1993.
- 44    A. Chi-Chih Yao. Some complexity questions related to distributive computing. In *Proceedings of the eleventh annual ACM symposium on Theory of computing*, pages 209–213, 1979.



# Relative Lipschitzness in Extragradient Methods and a Direct Recipe for Acceleration

**Michael B. Cohen**

Massachusetts Institute of Technology, Cambridge, MA, USA  
micochen@mit.edu

**Aaron Sidford**

Stanford University, CA, USA  
sidford@stanford.edu

**Kevin Tian**

Stanford University, CA, USA  
kjtian@stanford.edu

---

## Abstract

We show that standard extragradient methods (i.e. mirror prox [26] and dual extrapolation [28]) recover optimal accelerated rates for first-order minimization of smooth convex functions. To obtain this result we provide fine-grained characterization of the convergence rates of extragradient methods for solving monotone variational inequalities in terms of a natural condition we call *relative Lipschitzness*. We further generalize this framework to handle local and randomized notions of relative Lipschitzness and thereby recover rates for box-constrained  $\ell_\infty$  regression based on area convexity [34] and complexity bounds achieved by accelerated (randomized) coordinate descent [5, 29] for smooth convex function minimization.

**2012 ACM Subject Classification** Mathematics of computing → Convex optimization

**Keywords and phrases** Variational inequalities, minimax optimization, acceleration,  $\ell_\infty$  regression

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.62

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/2011.06572>.

**Funding** Researchers supported in part by Microsoft Research Faculty Fellowship, NSF CAREER Award CCF-1844855, NSF Grant CCF-1955039, a PayPal research gift, and a Sloan Research Fellowship.

**Acknowledgements** The existence of an extragradient algorithm in the primal-dual formulation of smooth minimization directly achieving accelerated rates is due to discussions with the first author, Michael B. Cohen. The second and third authors are indebted to him, and this work is dedicated in his memory. We also thank our collaborators in concurrent works, Yair Carmon and Yujia Jin, for many helpful and encouraging conversations throughout the duration of this project, Jelena Diakonikolas, Jonathan Kelner, and Jonah Sherman for helpful conversations, and anonymous reviewers for multiple helpful comments on earlier versions of the paper.

## 1 Introduction

We study the classic extragradient algorithms of mirror prox [26] and dual extrapolation [28] for solving variational inequalities (VIs) in monotone operators. This family of problems includes convex optimization and finding the saddle point of a convex-concave game. Due to applications of the latter to adversarial and robust training, extragradient methods have received significant recent attention in the machine learning community, see e.g. [12, 24, 16]. Further, extragradient methods have been the subject of increasing study by the theoretical computer science and optimization communities due to recent extragradient-based runtime improvements for problems including maximum flow [34] and zero-sum games [10, 11].



© Michael B. Cohen, Aaron Sidford, and Kevin Tian;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 62; pp. 62:1–62:18



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Given a Lipschitz monotone operator and a bounded strongly-convex regularizer, mirror prox [26] and dual extrapolation [28] achieve  $O(T^{-1})$  regret for solving the associated VI after  $T$  iterations. This rate is worst-case optimal when the Lipschitzness of the operator and strong convexity of the regularizer are with respect to the Euclidean norm [30]. However, in certain structured problems related to VIs, alternative analyses and algorithms can yield improved rates. For instance, when minimizing a smooth convex function (i.e. one with a Lipschitz gradient), it is known that accelerated rates of  $O(T^{-2})$  are attainable, improving upon the standard  $O(T^{-1})$  extragradient rate for the naive associated VI. Further, algorithms inspired by extragradient methods have been developed recovering the  $O(T^{-2})$  rate [13, 37].

Additionally, alternative analyses of extragradient methods, such as optimism [32] and area convexity [34] have shown that under assumptions beyond a Lipschitz operator and a strongly convex regularizer, improved rates can be achieved. These works leveraged modified algorithms which run efficiently under such non-standard assumptions. Further, the area convexity-based methods of [34] have had a number of implications, including faster algorithms for  $\ell_\infty$  regression, maximum flow and multicommodity flow [34] as well as improved parallel algorithms for work-efficient positive linear programming [8] and optimal transport [17].

In this work we seek a better understanding of these structured problems and the somewhat disparate-seeming analyses and algorithms for solving them. We ask, *are the algorithmic changes enabling these results necessary? Can standard mirror prox and dual extrapolation be leveraged to obtain these results? Can we unify analyses for these problems, and further clarify the relationship between acceleration, extragradient methods, and primal-dual methods?*

Towards addressing these questions, we provide a general condition, which we term *relative Lipschitzness* (cf. Definition 1), and analyze the convergence of mirror prox and dual extrapolation for a monotone relatively Lipschitz operator.<sup>1</sup> This condition is derived directly from the standard analysis of the methods and is stated in terms of a straightforward relationship between the operator  $g$  and the regularizer  $r$  which define the algorithm. Our condition is inspired by both area convexity as well as the “relative smoothness” condition in convex optimization [6, 23], and can be thought of as a generalization of the latter to variational inequalities (see Lemma 3). Further, through this analysis we show that standard extragradient methods directly yield accelerated rates for smooth minimization and recover the improved rates of [34] for box-constrained  $\ell_\infty$  regression, making progress on the questions outlined above. We also show our methods recover certain randomized accelerated rates and have additional implications, outlined below.

**Extragradient methods directly yield acceleration.** In Section 4, we show that applying algorithms of [26, 28] to a minimax formulation of  $\min_{x \in \mathbb{R}^d} f(x)$ , when  $f$  is smooth and strongly convex, yields accelerated rates when analyzed via relative Lipschitzness. Specifically, we consider the following problem, termed the *Fenchel game* in [37]:

$$\min_{x \in \mathbb{R}^d} \max_{y \in \mathbb{R}^d} \langle y, x \rangle - f^*(y), \quad (1)$$

<sup>1</sup> A somewhat similarly-named property appeared in [22], which also studied mirror descent algorithms under relaxed conditions; their property  $\|g(x)\|_*^2 \leq \frac{MV_x(y)}{\|y-x\|^2}$  for all  $x, y$ , is different than the one we propose. Further, during the preparation of this work, the relative Lipschitzness condition we propose was also independently stated in [36] (unknownst to the authors of this paper until recently). However, the work [36] does not derive the various consequences of relative Lipschitzness contained in this work (e.g. recovery of acceleration and randomized acceleration, as well as applications of area convexity).

and show that when  $f$  is  $\mu$ -strongly convex and  $L$ -smooth,  $O(\sqrt{L/\mu})$  iterations of either mirror prox [26] or dual extrapolation [28] produces an average iterate which halves the function error of  $f$ . By repeated application, this yields an accelerated linear rate of convergence and the optimal  $O(T^{-2})$  rates for non-strongly convex, smooth function minimization by a reduction [4]. Crucially, to attain this rate we give a sharpened bound on the relative Lipschitzness of the gradient operator of (1) with respect to a primal-dual regularizer.

Our result advances a recent line of research, [1, 2, 13], on applying primal-dual analyses to shed light on the mysterious nature of acceleration. Specifically, [1, 2] show that the classical algorithm of [27] can be rederived via applying primal-dual “optimistic” dynamics, inspired by the framework of [32]. Further, [13] showed that an appropriate discretization of dynamics inspired by extragradient algorithms yields an alternative accelerated algorithm. While these results clarify the primal-dual nature of acceleration, additional tuning is ultimately required to obtain their final algorithms and analysis. We obtain acceleration as a direct application of known frameworks, i.e. standard mirror prox and dual extrapolation, applied to the formulation (1), and hope this helps demystify acceleration.

In the full version of the paper, we further show that analyzing extragradient methods tailored to strongly monotone operators via relative Lipschitzness, and applying this more fine-grained analysis to a variant of the objective (1), also yields an accelerated linear rate of convergence. The resulting proof strategy extends readily to accelerated minimization of smooth and strongly convex functions in general norms, as we discuss at the end of Section 4, and we believe it may be of independent interest.

Finally, we remark that there has been documented difficulty in accelerating the minimization of relatively smooth functions [15]; this was also explored more formally by [14]. It is noted in [15], as well as suggested in others (e.g. in the development of area convexity [34]) that this discrepancy may be due to acceleration fundamentally requiring conditions on relationships between groups of three points, rather than two. Our work, which presents an alternative three-point condition yielding accelerated rates, sheds light on this phenomenon and we believe it is an interesting future direction to explore the relationships between our condition and other alternatives in the literature which are known to yield acceleration.

**Area convexity for bilinear box-simplex games.** In Section 5, we draw a connection between relative Lipschitzness and the notion of an “area convex” regularizer, proposed by [34]. Area convexity is a property which weakens strong convexity, but is suitable for extragradient algorithms with a linear operator. It was introduced in the context of solving a formulation of approximate undirected maximum flow via box-constrained  $\ell_\infty$  regression, or more generally approximating bilinear games between a box variable and a simplex variable. The algorithm of [34] applied to bilinear games was a variant of standard extragradient methods and analyzed via area convexity, which was proven via solving a subharmonic partial differential equation. We show that mirror prox, as analyzed by a local variant of relative Lipschitzness, yields the same rate of convergence as implied by area convexity, for box-simplex games. Our proof of this rate is straightforward and based on a simple Cauchy-Schwarz argument after demonstrating local stability of iterates.

**Randomized extragradient methods via local variance reduction.** In general, the use of stochastic operator estimates in the design of extragradient algorithms for solving general VIs is not as well-understood as their use in the special case of convex function minimization. The best-known known stochastic methods for solving VIs [19] with bounded-variance stochastic estimators obtain  $O(T^{-1/2})$  rates of convergence; this is by necessity, from known

classical lower bounds on the rate of the special case of stochastic convex optimization [25]. Towards advancing the randomized extragradient toolkit, we ask: when can improved  $O(T^{-1})$  rates of convergence be achieved by stochastic algorithms for solving specific VIs and fine-grained bounds on estimator variance (i.e. more local notions of variance)? This direction is inspired by analogous results in convex optimization, where reduced-variance and accelerated rates have been obtained, matching and improving upon their deterministic counterparts [18, 33, 3, 20, 29, 5].

For the special case of bilinear games, this question was recently addressed by the works [31, 10], using proximal reductions to attain improved rates. In this work, we give a framework for direct stochastic extragradient method design bypassing the variance bottleneck limiting prior algorithms to a  $O(T^{-1/2})$  rate of convergence for problems with block-separable structure. We identify a particular criterion of randomized operators used in the context of extragradient algorithms (cf. Proposition 12) which enables  $O(T^{-1})$  rates of convergence. Our approach is a form of “local variance reduction”, where estimators in an iteration of the method share a random seed and we take expectations over the whole iteration in the analysis. Our improved estimator design exploits the separable structure of the problem; it would be interesting to design a more general variance reduction framework for randomized extragradient methods.

Formally, we apply our local variance reduction framework in Section 6 to show that an instance of our new randomized extragradient methods recover acceleration for coordinate-smooth functions, matching the known tight rates of [5, 29]. Along the way, we give a variation of relative Lipschitzness capturing an analogous property between a locally variance-reduced randomized gradient estimator and a regularizer, which we exploit to obtain our runtime. We note that a similar approach was taken in [35] to obtain faster approximate maximum flow algorithms in the bilinear minimax setting; here, we generalize this strategy and give conditions under which our variance reduction technique obtains improved rates for extragradient methods more broadly.

**Additional contributions.** A minor contribution of our framework is that we show, in the full version of the paper, that relative Lipschitzness implies new rates for minimax convex-concave optimization, taking a step towards closing the gap with lower bounds with *fine-grained* dependence on problem parameters. Under operator-norm bounds on blocks of the Hessian of a convex-concave function, as well as blockwise strong convexity assumptions, [38] showed a lower bound on the convergence rate to obtain an  $\epsilon$ -approximate saddle point. When the blockwise operator norms of the Hessian are roughly equal, [21] gave an algorithm matching the lower bound up to a polylogarithmic factor, using an alternating scheme repeatedly calling an accelerated proximal point reduction. Applying our condition with a strongly monotone variant of the mirror prox algorithm of [26] yields a new fine-grained rate for minimax optimization, improving upon the runtime of [21] for a range of parameters. Our algorithm is simple and the analysis follows directly from a tighter relative Lipschitzness bound; we note the same result can also be obtained by considering an operator norm bound of the problem after a rescaling of space, but we include this computation because it is a straightforward implication of our condition.

Finally, in the full version, we also discuss the relation of relative Lipschitzness to another framework for analyzing extragradient methods: namely, we note that our proof of the sufficiency of relative Lipschitzness recovers known bounds for optimistic mirror descent [32].

## 2 Notation

**General notation.** Variables are in  $\mathbb{R}^d$  unless otherwise noted.  $e_i$  is the  $i^{\text{th}}$  standard basis vector.  $\|\cdot\|$  denotes an arbitrary norm; the dual norm is  $\|\cdot\|_*$ , defined as  $\|x\|_* := \max_{\|y\| \leq 1} y^\top x$ . For a variable on two blocks  $z \in \mathcal{X} \times \mathcal{Y}$ , we refer to the blocks by  $z^x$  and  $z^y$ . We denote the domain of  $f : \mathbb{R}^d \rightarrow \mathbb{R}$  by  $\mathcal{X}$ ; when unspecified,  $\mathcal{X} = \mathbb{R}^d$ . When  $f$  is clear from context,  $x^*$  is any minimizing argument. We call any  $x$  with  $f(x) \leq f(x^*) + \epsilon$  an  $\epsilon$ -approximate minimizer.

**Bregman divergences.** The Bregman divergence induced by convex  $r$  is

$$V_x^r(y) := r(y) - r(x) - \langle \nabla r(x), y - x \rangle.$$

The Bregman divergence is always nonnegative, and convex in its argument. We define the following proximal operation with respect to a divergence from point  $z$ .

$$\text{Prox}_x^r(g) := \operatorname{argmin}_y \{ \langle g, y \rangle + V_x^r(y) \}. \quad (2)$$

**Functions.** We say  $f$  is  $L$ -smooth in  $\|\cdot\|$  if  $\|\nabla f(x) - \nabla f(y)\|_* \leq L \|x - y\|$ , or equivalently  $f(y) \leq f(x) + \langle \nabla f(x), y - x \rangle + \frac{L}{2} \|y - x\|^2$  for  $x, y \in \mathcal{X}$ . If  $f$  is twice-differentiable, equivalently  $y^\top \nabla^2 f(x) y \leq L \|y\|^2$ . We say differentiable  $f$  is  $\mu$ -strongly convex if for some  $\mu \geq 0$ ,  $f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle + \frac{\mu}{2} \|y - x\|^2$  for  $x, y \in \mathcal{X}$ . We also say  $f$  is  $\mu$ -strongly convex with respect to a distance-generating function  $r$  if  $V_x^f(y) \geq \mu V_x^r(y)$  for all  $x, y \in \mathcal{X}$ . Further, we use standard results from convex analysis throughout, in particular facts about Fenchel duality, and defer these definitions and proofs to the full version of the paper.

**Saddle points.** We call function  $h(x, y)$  of two variables *convex-concave* if its restrictions to  $x$  and  $y$  are convex and concave respectively. We call  $(x, y)$  an  $\epsilon$ -approximate saddle point if  $\max_{y'} \{h(x, y')\} - \min_{x'} \{h(x', y)\} \leq \epsilon$ . We equip any differentiable convex-concave function with gradient operator  $g(x, y) := (\nabla_x h(x, y), -\nabla_y h(x, y))$ .

**Monotone operators.** We call operator  $g : \mathcal{Z} \rightarrow \mathcal{Z}^*$  monotone if  $\langle g(w) - g(z), w - z \rangle \geq 0$  for all  $w, z \in \mathcal{Z}$ . Examples include the gradient of a convex function and gradient operator of a convex-concave function. We call  $g$   $m$ -strongly monotone with respect to  $r$  if  $\langle g(w) - g(z), w - z \rangle \geq m (V_w^r(z) + V_z^r(w))$ . We call  $z^* \in \mathcal{Z}$  the solution to the variational inequality (VI) in a monotone operator  $g$  if  $\langle g(z^*), z^* - z \rangle \leq 0$  for all  $z \in \mathcal{Z}$ .<sup>2</sup> Examples include the minimizer of a convex function and the saddle point of a convex-concave function.

## 3 Extragradient convergence under relative Lipschitzness

We give a brief presentation of mirror prox [26], and a convergence analysis under relative Lipschitzness. Our results also hold for dual extrapolation [28], which can be seen as a “lazy” version of mirror prox updating a state in dual space (see [9]); we defer details to the full version of the paper.

► **Definition 1** (Relative Lipschitzness). For convex  $r : \mathcal{Z} \rightarrow \mathbb{R}$ , we call operator  $g : \mathcal{Z} \rightarrow \mathcal{Z}^*$   $\lambda$ -relatively Lipschitz with respect to  $r$  if for every three  $z, w, u \in \mathcal{Z}$ ,

$$\langle g(w) - g(z), w - u \rangle \leq \lambda (V_z^r(w) + V_w^r(u))$$

<sup>2</sup> This is also known as a “strong solution”. A “weak solution” is a  $z^*$  with  $\langle g(z), z^* - z \rangle \leq 0$  for all  $z$ .

■ **Algorithm 1** MIRROR-PROX( $z_0, T$ ): Mirror prox [26].

---

**Input:** Distance generating  $r$ ,  $\lambda$ -relatively Lipschitz monotone  $g : \mathcal{Z} \rightarrow \mathcal{Z}^*$ , initial point  $z_0 \in \mathcal{Z}$   
**for**  $0 \leq t < T$  **do**  
     $w_t \leftarrow \text{Prox}_{z_t}^r(\frac{1}{\lambda}g(z_t))$   
     $z_{t+1} \leftarrow \text{Prox}_{z_t}^r(\frac{1}{\lambda}g(w_t))$   
**end for**

---

Definition 1 can be thought of as an alternative to a natural nonlinear analog of the area convexity condition of [34] displayed below:

$$\langle g(w) - g(z), w - u \rangle \leq \lambda \left( r(z) + r(w) - r(u) - 3r\left(\frac{z + w + u}{3}\right) \right).$$

Our proposed alternative is well-suited for the standard analyses of extragradient methods such as mirror prox and dual extrapolation. For the special case of bilinear minimax problems in a matrix  $A$ , the left hand side of Definition 1 measures the area of a triangle in a geometry induced by  $A$ .

Relative Lipschitzness encapsulates the more standard assumptions that  $g$  is Lipschitz and  $r$  is strongly convex (Lemma 2), as well as the more recent assumptions that  $f$  is convex and relatively smooth with respect to  $r$  [6, 23] (Lemma 3).

► **Lemma 2.** *If  $g$  is  $L$ -Lipschitz and  $r$  is  $\mu$ -strongly convex in  $\|\cdot\|$ ,  $g$  is  $L/\mu$ -relatively Lipschitz with respect to  $r$ .*

**Proof.** By Cauchy-Schwarz, Lipschitzness of  $g$ , and strong convexity of  $r$ ,

$$\begin{aligned} \langle g(w) - g(z), w - u \rangle &\leq \|g(w) - g(z)\|_* \|w - u\| \leq L \|w - z\| \|w - u\| \\ &\leq \frac{L}{2} (\|w - z\|^2 + \|w - u\|^2) \leq \frac{L}{\mu} (V_z^r(w) + V_w^r(u)). \end{aligned} \quad \blacktriangleleft$$

► **Lemma 3.** *If  $f$  is  $L$ -relatively smooth with respect to  $r$ , i.e.  $V_x^f(y) \leq LV_x^r(y)$  for all  $x$  and  $y$ , then  $g$ , defined by  $g(x) := \nabla f(x)$  for all  $x$ , is  $L$ -relatively Lipschitz with respect to  $r$ .*

**Proof.** By assumption of relative smoothness of  $f$  and the definition of divergence,

$$\begin{aligned} L(V_z^r(w) + V_w^r(u)) &\geq V_z^f(w) + V_w^f(u) \\ &= f(w) - [f(z) + \nabla f(z)^\top (w - z)] + f(u) - [f(w) + \nabla f(w)^\top (u - w)] \\ &= V_z^f(u) - \nabla f(z)^\top (z - u) - \nabla f(z)^\top (w - z) + \nabla f(w)^\top (w - u) \\ &= V_z^f(u) + \langle g(w) - g(z), u - z \rangle. \end{aligned}$$

The result follows from the fact that  $V_z^f(u) \geq 0$  by convexity of  $f$ . ◀

We now give an analysis of Algorithm 1 showing the average “regret”  $\langle g(w_t), w_t - u \rangle$  of iterates decays at a  $O(T^{-1})$  rate. This strengthens Lemma 3.1 of [26].

► **Proposition 4.** *The iterates  $\{w_t\}$  of Algorithm 1 satisfy for all  $u \in \mathcal{Z}$ ,*

$$\sum_{0 \leq t < T} \langle g(w_t), w_t - u \rangle \leq \lambda V_{z_0}^r(u).$$

**Proof.** First-order optimality conditions of  $w_t, z_{t+1}$  with respect to  $u$  imply

$$\begin{aligned} \frac{1}{\lambda} \langle g(z_t), w_t - z_{t+1} \rangle &\leq V_{z_t}^r(z_{t+1}) - V_{w_t}^r(z_{t+1}) - V_{z_t}^r(w_t), \\ \frac{1}{\lambda} \langle g(w_t), z_{t+1} - u \rangle &\leq V_{z_t}^r(u) - V_{z_{t+1}}^r(u) - V_{z_t}^r(z_{t+1}). \end{aligned} \quad (3)$$

Adding and manipulating gives, via relative Lipschitzness (Definition 1),

$$\begin{aligned} \frac{1}{\lambda} \langle g(w_t), w_t - u \rangle &\leq V_{z_t}^r(u) - V_{z_{t+1}}^r(u) + \frac{1}{\lambda} \langle g(w_t) - g(z_t), w_t - z_{t+1} \rangle - V_{w_t}^r(z_{t+1}) - V_{z_t}^r(w_t) \\ &\leq V_{z_t}^r(u) - V_{z_{t+1}}^r(u). \end{aligned} \quad (4)$$

Finally, summing and telescoping (4) yields the desired conclusion.  $\blacktriangleleft$

We briefly comment on how to use Proposition 4 to approximately solve convex-concave games in a function  $f(x, y)$ . By applying convexity and concavity appropriately to the regret guarantee (and dividing by  $T$ , the iteration count), one can replace the left hand side of the guarantee with the duality gap of an average point  $\bar{w}$  against a point  $u$ , namely  $f(w^x, u^y) - f(u^x, w^y)$ . By maximizing the right hand side over  $u$ , this can be converted into an overall duality gap guarantee. For some of our applications in following sections,  $u$  will be some fixed point (rather than a best response) and the regret statement will be used in a more direct manner to prove guarantees.

## 4 Acceleration via relative Lipschitzness

We show that directly applying Algorithm 1 to the optimization problem (1) recovers an accelerated rate for first-order convex function minimization (for simplicity, we focus on the  $\ell_2$  norm here; our methods extend to general norms, discussed in the full version of the paper). Our main technical result, Lemma 5, shows the gradient operator of (1) is relatively Lipschitz in the natural regularizer induced by  $f$ , which combined with Proposition 4 gives our main result, Theorem 7. Crucially, our method regularizes the dual variable with  $f^*$ , the Fenchel dual of  $f$ , which we show admits efficient implementation, allowing us to obtain our improved bound on the relative Lipschitzness parameter.

► **Lemma 5** (Relative Lipschitzness for the Fenchel game). *Let  $f : \mathbb{R}^d \rightarrow \mathbb{R}$  be  $L$ -smooth and  $\mu$ -strongly convex in the Euclidean norm  $\|\cdot\|_2$ . Let  $g(x, y) = (y, \nabla f^*(y) - x)$  be the gradient operator of the convex-concave problem (1), and define the distance-generating function  $r(x, y) := \frac{\mu}{2} \|x\|_2^2 + f^*(y)$ . Then,  $g$  is  $1 + \sqrt{\frac{L}{\mu}}$ -relatively Lipschitz with respect to  $r$ .*

**Proof.** Consider three points  $z = (z^x, z^y)$ ,  $w = (w^x, w^y)$ ,  $u = (u^x, u^y)$ . By direct calculation,

$$\langle g(w) - g(z), w - u \rangle = \langle w^y - z^y, w^x - u^x \rangle + \langle -w^x + z^x + \nabla f^*(w^y) - \nabla f^*(z^y), w^y - u^y \rangle. \quad (5)$$

By Cauchy-Schwarz and  $L^{-1}$ -strong convexity of  $f^*$  respectively, we have

$$\begin{aligned} \langle w^y - z^y, w^x - u^x \rangle + \langle z^x - w^x, w^y - u^y \rangle &\leq \|w^y - z^y\|_2 \|w^x - u^x\|_2 + \|z^x - w^x\|_2 \|w^y - u^y\|_2 \\ &\leq \sqrt{\frac{L}{\mu}} \left( \frac{\mu}{2} \|w^x - z^x\|_2^2 + \frac{\mu}{2} \|w^x - u^x\|_2^2 + \frac{1}{2L} \|w^y - z^y\|_2^2 + \frac{1}{2L} \|w^y - u^y\|_2^2 \right) \\ &\leq \sqrt{\frac{L}{\mu}} (V_z^r(w) + V_w^r(u)). \end{aligned} \quad (6)$$



The second line used Young's inequality twice. Furthermore, by convexity of  $f^*$  from  $z^y$  to  $u^y$ ,

$$\begin{aligned}
& \langle \nabla f^*(w^y) - \nabla f^*(z^y), w^y - u^y \rangle \\
&= \langle \nabla f^*(z^y), u^y - z^y \rangle - \langle \nabla f^*(w^y), u^y - w^y \rangle - \langle \nabla f^*(z^y), w^y - z^y \rangle \\
&\leq f^*(u^y) - f^*(z^y) - \langle \nabla f^*(w^y), u^y - w^y \rangle - \langle \nabla f^*(z^y), w^y - z^y \rangle \\
&= V_{z^y}^{f^*}(w^y) + V_{w^y}^{f^*}(u^y) \leq V_z^r(w) + V_w^r(u).
\end{aligned} \tag{7}$$

The last inequality used separability of  $r$  and nonnegativity of divergences. Summing the bounds (6) and (7) and recalling (5) yields the conclusion, where we use Definition 1.  $\blacktriangleleft$

We also state a convenient fact about the form our iterates take.

► **Lemma 6.** *In the setting of Lemma 5, let  $z_t = (x_t, y_t)$ ,  $w_t = (x_{t+\frac{1}{2}}, y_{t+\frac{1}{2}})$  be iterates produced by running Algorithm 1 on the pair  $g, r$ . Suppose  $y_0 = \nabla f(v_0)$  for some  $v_0$ . Then,  $y_{t+\frac{1}{2}}$  and  $y_{t+1}$  can be recursively expressed as  $y_{t+\frac{1}{2}} = \nabla f(v_{t+\frac{1}{2}})$ ,  $y_{t+1} = \nabla f(v_{t+1})$ , for*

$$v_{t+\frac{1}{2}} \leftarrow v_t + \frac{1}{\lambda}(x_t - v_t), \quad v_{t+1} \leftarrow v_t + \frac{1}{\lambda}\left(x_{t+\frac{1}{2}} - v_{t+\frac{1}{2}}\right).$$

**Proof.** We prove this inductively; consider some iteration  $t$ . Assuming  $y_t = \nabla f(v_t)$ , by definition

$$\begin{aligned}
y_{t+\frac{1}{2}} &= \operatorname{argmin}_y \left\{ \left\langle \frac{1}{\lambda}(\nabla f^*(y_t) - x_t), y \right\rangle + V_{y_t}^{f^*}(y) \right\} \\
&= \operatorname{argmax}_y \left\{ \left\langle \frac{1}{\lambda}(x_t - v_t) + v_t, y \right\rangle - f^*(y) \right\} = \nabla f\left(v_t + \frac{1}{\lambda}(x_t - v_t)\right).
\end{aligned}$$

Here, we used standard facts about convex conjugates. A similar argument shows that we can compute implicitly  $y_{t+1} = \nabla f(v_t + \frac{1}{\lambda}(x_{t+\frac{1}{2}} - v_{t+\frac{1}{2}}))$ .  $\blacktriangleleft$

We now prove Theorem 7, i.e. that we can halve function error in  $O\left(\sqrt{\frac{L}{\mu}}\right)$  iterations of Algorithm 1. Simply iterating Theorem 7 yields a linear rate of convergence for smooth, strongly convex functions, yielding an  $\epsilon$ -approximate minimizer in  $O\left(\sqrt{\frac{L}{\mu}} \log \frac{f(x_0) - f(x^*)}{\epsilon}\right)$  iterations.

► **Theorem 7.** *In the setting of Lemma 5, run  $T \geq 4\lambda$  iterations of Algorithm 1 initialized at  $z_0 = (x_0, \nabla f(x_0))$  on the pair  $g, r$  with  $\lambda = 1 + \sqrt{\frac{L}{\mu}}$ , and define*

$$\bar{v} = \frac{1}{T} \sum_{0 \leq t < T} v_{t+\frac{1}{2}} \text{ where } w_t = \left(x_{t+\frac{1}{2}}, \nabla f(v_{t+\frac{1}{2}})\right).$$

*Then we have  $f(\bar{v}) - f(x^*) \leq \frac{1}{2}(f(x_0) - f(x^*))$ , where  $x^*$  minimizes  $f$ .*

**Proof.** First, we remark that this form of  $w_t$  follows from Lemma 6, and correctness of  $\lambda$  follows from Lemma 5. By an application of Proposition 4, letting  $u = (x^*, \nabla f(x^*))$ ,

$$\begin{aligned}
\frac{1}{T} \sum_{0 \leq t < T} \langle g(w_t), w_t - u \rangle &\leq \frac{\lambda}{T} \cdot V_{z_0}^r(u) \leq \frac{1}{4} \left( \frac{\mu}{2} \|x_0 - x^*\|_2^2 + V_{\nabla f(x_0)}^{f^*}(\nabla f(x^*)) \right) \\
&= \frac{1}{4} \left( \frac{\mu}{2} \|x_0 - x^*\|_2^2 + f(x_0) - f(x^*) \right) \leq \frac{1}{2} (f(x_0) - f(x^*)).
\end{aligned}$$

The second line used the definition of divergence in  $f^*$  and strong convexity of  $f$ , which implies  $f(x_0) \geq f(x^*) + \frac{\mu}{2} \|x_0 - x^*\|_2^2$ . Moreover, by the definition of  $g$  and  $\nabla f(x^*) = 0$ ,

$$\begin{aligned} \frac{1}{T} \sum_{0 \leq t < T} \langle g(w_t), w_t - u \rangle &= \frac{1}{T} \sum_{0 \leq t < T} \langle \nabla f(v_{t+\frac{1}{2}}), x_{t+\frac{1}{2}} - x^* \rangle + \langle v_{t+\frac{1}{2}} - x_{t+\frac{1}{2}}, \nabla f(v_{t+\frac{1}{2}}) \rangle \\ &\geq \frac{1}{T} \sum_{0 \leq t < T} f(v_{t+\frac{1}{2}}) - f(x^*) \geq f(\bar{v}) - f(x^*). \end{aligned}$$

The last line used convexity twice. Combining these two derivations yields the conclusion. ◀

For convenience, we state the full algorithm of Theorem 7 as Algorithm 2.

■ **Algorithm 2** EG-ACCEL( $x_0, \epsilon$ ): Extragradient accelerated smooth minimization.

**Input:**  $x_0 \in \mathbb{R}^d$ ,  $f$   $L$ -smooth and  $\mu$ -strongly convex in  $\|\cdot\|_2$ , and  $\epsilon_0 \geq f(x_0) - f(x^*)$

**Output:**  $\epsilon$ -approximate minimizer of  $f$

$\lambda \leftarrow 1 + \sqrt{L/\mu}$ ,  $x^{(0)} \leftarrow x_0$ ,  $T \leftarrow 4\lceil\lambda\rceil$ ,  $K \leftarrow \lceil\log_2 \frac{\epsilon_0}{\epsilon}\rceil$

**for**  $0 \leq k < K$  **do**

$x_0 \leftarrow x^{(k)}$ ,  $v_0 \leftarrow x_0$

**for**  $0 \leq t < T$  **do**

$x_{t+\frac{1}{2}} \leftarrow x_t - \frac{1}{\mu\lambda} \nabla f(v_t)$  and  $v_{t+\frac{1}{2}} \leftarrow v_t + \frac{1}{\lambda}(x_t - v_t)$

$x_{t+1} \leftarrow x_t - \frac{1}{\mu\lambda} \nabla f(v_{t+\frac{1}{2}})$  and  $v_{t+1} \leftarrow v_t + \frac{1}{\lambda}(x_{t+\frac{1}{2}} - v_{t+\frac{1}{2}})$

**end for**

$x^{(k+1)} \leftarrow \frac{1}{T} \sum_{0 \leq t < T} v_{t+\frac{1}{2}}$

**end for**

**return**  $x^{(K)}$

In the full version of the paper, we give an alternative proof of acceleration leveraging relative Lipschitzness, as well as a variant of extragradient methods suited for strongly monotone operators, by applying these tools to the saddle point problem (to be contrasted with (1))

$$\min_{x \in \mathbb{R}^d} f(x) = \min_{x \in \mathbb{R}^d} \max_{y \in \mathbb{R}^d} \frac{\mu}{2} \|x\|_2^2 + \langle y, x \rangle - h^*(y), \text{ where } h(x) := f(x) - \frac{\mu}{2} \|x\|_2^2.$$

This alternative proof strategy readily generalizes the accelerated rate of Theorem 7 to general norms. While the rates attained by this alternative method are slightly less sharp (losing a  $\frac{L}{\mu}$  factor in the logarithm) when compared to Theorem 7, the analysis is arguably simpler. This is in the sense that our alternative strategy shows a potential function decreases at a linear rate in every iteration, rather than requiring  $O(\sqrt{L/\mu})$  iterations to halve it.

## 5 Area convexity rates for box-simplex games via relative Lipschitzness

In this section, we show that a local variant of Definition 1 recovers the improved convergence rate achieved by [34] for box-constrained  $\ell_\infty$ -regression, and more generally box-simplex bilinear games. Specifically, we will use the following result, a simple extension to Proposition 4 which states that relative Lipschitzness only must hold with respect to triples of algorithm iterates.

► **Corollary 8.** *Suppose Algorithm 1 is run with a monotone operator  $g$  and a distance generating  $r$  satisfying, for all iterations  $t$ ,*

$$\langle g(w_t) - g(z_t), w_t - z_{t+1} \rangle \leq \lambda (V_{z_t}^r(w_t) + V_{w_t}(z_{t+1})). \quad (8)$$

*Then, the conclusion of Proposition 4 holds.*

**Proof.** Observe that the only applications of relative Lipschitzness in the proof of Proposition 4 are of the form (8) (namely, in (4)). Thus, the same conclusion still holds. ◀

We use Corollary 8 to give an alternative algorithm and analysis recovering the rates implied by the use of area convexity in [34], for box-simplex games, which we now define.

► **Definition 9 (Box-simplex game).** Let  $A \in \mathbb{R}^{m \times n}$  be a matrix and let  $b \in \mathbb{R}^m$ ,  $c \in \mathbb{R}^n$  be vectors. The associated box-simplex game, and its induced monotone operator  $g$ , are

$$\min_{x \in [-1, 1]^n} \max_{y \in \Delta^m} f(x, y) := y^\top Ax - \langle b, y \rangle + \langle c, x \rangle, \quad g(x, y) := (A^\top y + c, b - Ax). \quad (9)$$

Here,  $\Delta^m := \{y \in \mathbb{R}_{\geq 0}^m : \sum_{i \in [m]} y_i = 1\}$  is the nonnegative probability simplex in  $m$  dimensions.

By a simple reduction that at most doubles the size of the input (stacking  $A$ ,  $b$  with negated copies, cf. Section 3.1 of [35]), Definition 9 is a generalization of the box-constrained  $\ell_\infty$ -regression problem

$$\min_{x \in [-1, 1]^m} \|Ax - b\|_\infty.$$

The work of [34] proposed a variant of extragradient algorithms, based on taking primal-dual proximal steps in the following regularizer:<sup>3</sup>

$$r(x, y) := y^\top |A|(x^2) + 10 \|A\|_{\infty \rightarrow \infty} \sum_{i \in [m]} y_i \log y_i. \quad (10)$$

Here,  $|A|$  is the entrywise absolute value of  $A$ . The convergence rate of this algorithm was proven in [34] via an analysis based on “area convexity” of the pair  $(g, r)$ , which required a somewhat sophisticated proof based on solving a partial differential equation over a triangle. We now show that the same rate can be obtained by the extragradient algorithms of [26, 28], and analyzed via local relative Lipschitzness (8).<sup>4</sup> We first make the following simplification without loss of generality.

► **Lemma 10.** For all  $x \in [-1, 1]^n$  the value of  $\max_{y \in \Delta^m} f(x, y)$  in (9) is unchanged if we remove all coordinates of  $b$  with  $b_i \geq \min_{i^* \in [m]} b_{i^*} + 2 \|A\|_{\infty \rightarrow \infty}$ , and the corresponding rows of  $A$ . Therefore, in designing an algorithm to solve (9) to additive error with linear pre-processing it suffices to assume that  $b_i \in [0, 2 \|A\|_{\infty \rightarrow \infty}]$  for all  $i \in [m]$ .

**Proof.** For any  $x \in [-1, 1]^n$ , letting  $i^* \in \operatorname{argmin}_{i \in [m]} b_i$  we have

$$\max_{y \in \Delta^m} y^\top (Ax - b) = \max_{i \in [m]} [Ax - b]_i \geq -\|A\|_{\infty \rightarrow \infty} \|x\|_\infty - \min_{i^* \in [m]} b_i \geq -\|A\|_{\infty \rightarrow \infty} - b_{i^*}.$$

However,  $[Ax - b]_i \leq \|A\|_{\infty \rightarrow \infty} - b_i$  for all  $i \in [m]$ . Consequently, any coordinate  $i \in [m]$  that satisfies  $b_i \geq b_{i^*} + 2 \|A\|_{\infty \rightarrow \infty}$  has  $[Ax - b]_i \leq [Ax - b]_{i^*}$  and the value of  $\max_{y \in \Delta^m} f(x, y)$  is unchanged if this entry of  $b_i$  and the corresponding row of  $A$  is removed. Further, note that  $\langle y, \mathbf{1} \rangle$  is a constant for all  $y \in \Delta^m$ . Consequently, in linear time we can remove all the coordinates  $i$  with  $b_i \geq \min_{i^* \in [m]} b_{i^*} + 2 \|A\|_{\infty \rightarrow \infty}$  and shift all the coordinates by an additive constant so that the minimum coordinate of a remaining  $b_i$  is 0 without affecting additive error of any  $x$ . ◀

<sup>3</sup> We let  $\|A\|_{\infty \rightarrow \infty} := \sup_{\|x\|_\infty=1} \|Ax\|_\infty$ , i.e. the  $\ell_\infty$  operator norm of  $A$  or max  $\ell_1$  norm of any row.

<sup>4</sup> Although our analysis suffices to recover the rate of [34] for  $\ell_\infty$  regression, the analysis of [34] is in some sense more robust (and possibly) more broadly applicable than ours, as it does not need to reason directly about how much the iterates vary in a step. Understanding or closing this gap is an interesting open problem.

We now prove our main result regarding the use of mirror prox to solve box-simplex games, using the regularizer analyzed (with a slightly different algorithm) in [34].

► **Theorem 11.** *Assume the preprocessing of Lemma 10 so that  $b \in [0, 2\|A\|_{\infty \rightarrow \infty}]^m$ . Consider running Algorithm 1 on the operator in (9) with  $\lambda = 3$ , using the regularizer in (10). The resulting iterates satisfy (8), and thus satisfy the conclusion of Proposition 4.*

**Proof.** Fix a particular iteration  $t$ . We first claim that the simplex variables  $w_t^y$  and  $z_{t+1}^y$  obey the following multiplicative stability property: entrywise,

$$w_t^y, z_{t+1}^y \in \left[ \frac{1}{2} z_t^y, 2z_t^y \right]. \quad (11)$$

We will give the proof for  $w_t^y$  as the proof for  $z_{t+1}^y$  follows from the same reasoning. Recall that

$$w_t = \operatorname{argmin}_{w \in \Delta^n \times [-1, 1]^m} \left\{ \left\langle \frac{1}{\lambda} g(z_t), w \right\rangle + V_{z_t}^r(w) \right\},$$

and therefore, defining  $(x)^2$  and  $(z_t^x)^2$  as the entrywise square of these vectors,

$$w_t^y = \operatorname{argmin}_{y \in \Delta^m} \langle \gamma_t^y, y \rangle + 10 \|A\|_{\infty \rightarrow \infty} \sum_{i \in [m]} y_i \log \frac{y_i}{[z_t^y]_i}$$

$$\text{where } \gamma_t^y := \frac{1}{\lambda} (b - Az_t^x) + |A| \left[ (x)^2 - (z_t^x)^2 \right].$$

Consequently, applying log and exp entrywise we have

$$w_t^y \propto \exp \left( \log z_t^y - \frac{1}{10 \|A\|_{\infty \rightarrow \infty}} \gamma_t^y \right).$$

This implies the desired (11), where we use that  $\|\gamma_t^y\|_{\infty} \leq 3\|A\|_{\infty \rightarrow \infty}$ , and  $\exp(0.6) \leq 2$ . Next, we have by a straightforward calculation (Lemma 3.4, [34] or Lemma 6, [17]) that

$$\nabla^2 r(x, y) \succeq \begin{pmatrix} \mathbf{diag}(|A_{:,j}|^\top y) & 0 \\ 0 & \|A\|_{\infty \rightarrow \infty} \mathbf{diag}\left(\frac{1}{y_i}\right) \end{pmatrix}. \quad (12)$$

By expanding the definition of Bregman divergence, we have

$$V_{z_t}^r(w_t) = \int_0^1 \int_0^\alpha \|w_t - z_t\|_{\nabla^2 r(z_t + \beta(w_t - z_t))}^2 d\beta d\alpha.$$

Fix some  $\beta \in [0, 1]$ , and let  $z_\beta := z_t + \beta(w_t - z_t)$ . Since the coordinates of  $z_\beta$  also satisfy the stability property (11), by the lower bound of (12), we have

$$\begin{aligned} \|w_t - z_t\|_{\nabla^2 r(z_\beta)}^2 &\geq \sum_{i \in [m], j \in [n]} |A_{ij}| \left( [z_\beta^y]_i [w_t^x - z_t^x]_j^2 + \frac{1}{[z_\beta^y]_i} [w_t^y - z_t^y]_i^2 \right) \\ &\geq \frac{1}{2} \sum_{i \in [m], j \in [n]} |A_{ij}| \left( [z_t^y]_i [w_t^x - z_t^x]_j^2 + \frac{1}{[z_t^y]_i} [w_t^y - z_t^y]_i^2 \right). \end{aligned}$$

By using a similar calculation to lower bound  $V_{w_t}^r(z_{t+1})$ , we have by Young's inequality the desired

$$\begin{aligned}
V_{z_t}^r(w_t) + V_{w_t}^r(z_{t+1}) &\geq \frac{1}{4} \sum_{i \in [m], j \in [n]} |A_{ij}| \left( [z_t^y]_i [w_t^x - z_t^x]_j^2 + \frac{1}{[z_t^y]_i} [w_t^y - z_t^y]_i^2 \right) \\
&\quad + \frac{1}{4} \sum_{i \in [m], j \in [n]} |A_{ij}| \left( [z_t^y]_i [w_t^x - z_{t+1}^x]_j^2 + \frac{1}{[z_t^y]_i} [w_t^y - z_{t+1}^y]_i^2 \right) \\
&\geq \frac{1}{3} \sum_{i \in [m], j \in [n]} A_{ij} ([w_t^y - z_t^y]_i [w_t^x - z_{t+1}^x]_j - [w_t^y - z_{t+1}^y]_i [w_t^x - z_t^x]_j) \\
&= \frac{1}{\lambda} \langle g(w_t) - g(z_t), w_t - z_{t+1} \rangle. \quad \blacktriangleleft
\end{aligned}$$

The range of the regularizer  $r$  is bounded by  $O(\|A\|_{\infty \rightarrow \infty} \log m)$ , and hence the iteration complexity to find an  $\epsilon$  additively-approximate solution to the box-simplex game is  $O(\frac{\|A\|_{\infty \rightarrow \infty} \log m}{\epsilon})$ . Finally, we comment that the iteration complexity of solving the subproblems required by extragradient methods in the regularizer  $r$  to sufficiently high accuracy is logarithmically bounded in problem parameters via a simple alternating minimization scheme proposed by [34]. Here, we note that the error guarantee e.g. Proposition 4 is robust up to constant factors to solving each subproblem to  $\epsilon$  additive accuracy, and appropriately using approximate optimality conditions (for an example of this straightforward extension, see Corollary 1 of [17]).

## 6 Randomized coordinate acceleration via expected relative Lipschitzness

We show relative Lipschitzness can compose with randomization. Specifically, we adapt Algorithm 2 to coordinate smoothness, recovering the accelerated rate first obtained in [5, 29]. We recall  $f$  is  $L_i$ -coordinate-smooth if its coordinate restriction is smooth, i.e.  $|\nabla_i f(x + ce_i) - \nabla_i f(x)| \leq L_i |c| \forall x \in \mathcal{X}, c \in \mathbb{R}$ ; for twice-differentiable coordinate smooth  $f$ ,  $\nabla_{ii}^2 f(x) \leq L_i$ .

Along the way, we build a framework for randomized extragradient methods via “local variance reduction” in Proposition 12. In particular, we demonstrate how for separable domains our technique can yield  $O(T^{-1})$  rates for stochastic extragradient algorithms, bypassing a variance barrier encountered by prior methods [19]. Throughout, let  $f : \mathbb{R}^d \rightarrow \mathbb{R}$  be  $L_i$ -smooth in coordinate  $i$ , and  $\mu$ -strongly convex in  $\|\cdot\|_2$ , and define the distance generating function  $r(x, y) = \frac{\mu}{2} \|x\|_2^2 + f^*(y)$ .

Our approach modifies that of Section 4 in the following ways. First, our iterates are defined via stochastic estimators which “share randomness” (use the same coordinate in both updates). Concretely, fix some iterate  $z_t = (x_t, \nabla f(v_t))$ . For a distribution  $\{p_i\}_{i \in [d]}$ , sample  $i \sim p_i$  and let

$$\begin{aligned}
g_i(z_t) &:= \left( \frac{1}{p_i} \nabla_i f(v_t), v_t - x_t \right), \quad w_t^{(i)} = \left( x_{t+\frac{1}{2}}^{(i)}, \nabla f(v_{t+\frac{1}{2}}) \right) := \text{Prox}_{z_t}^r \left( \frac{1}{\lambda} g_i(z_t) \right), \\
g_i(w_t^{(i)}) &:= \left( \frac{1}{p_i} \nabla_i f(v_{t+\frac{1}{2}}), v_{t+\frac{1}{2}} - \left( x_t + \frac{1}{p_i} \Delta_t^{(i)} \right) \right) \text{ for } \Delta_t^{(i)} := x_{t+\frac{1}{2}}^{(i)} - x_t, \\
z_{t+1}^{(i)} &= \left( x_{t+1}^{(i)}, \nabla f(v_{t+1}) \right) := \text{Prox}_{z_t}^r \left( \frac{1}{\lambda} g_i(w_t^{(i)}) \right).
\end{aligned} \tag{13}$$

By observation,  $g_i(z_t)$  is unbiased for  $g(z_t)$ ; however, the same cannot be said for  $g_i(w_t^{(i)})$ , as the random coordinate was used in the definition of  $w_t^{(i)}$ . Nonetheless, examining the proof of Proposition 4, we see that the conclusion

$$\langle g(\bar{w}_t), \bar{w}_t - u \rangle \leq V_{z_t}^r(u) - \mathbb{E} \left[ V_{z_{t+1}^{(i)}}^r(u) \right]$$

still holds for some point  $\bar{w}_t$ , as long as

$$\begin{aligned} \mathbb{E} \left[ \langle g_i(w_t^{(i)}), w_t^{(i)} - u \rangle \right] &= \langle g(\bar{w}_t), \bar{w}_t - u \rangle, \\ \mathbb{E} \left[ \langle g_i(w_t^{(i)}) - g_i(z_t), w_t^{(i)} - z_{t+1}^{(i)} \rangle \right] &\leq \lambda \mathbb{E} \left[ V_{z_t}^r(w_t^{(i)}) + V_{w_t^{(i)}}^r(z_{t+1}^{(i)}) \right]. \end{aligned} \quad (14)$$

We make this concrete in the following claim, a generalization of Proposition 4 which handles randomized operator estimates as well as an expected variant of relative Lipschitzness. We remark that as in Corollary 8, the second condition in (14) only requires relative Lipschitzness to hold for the iterates of the algorithm, rather than globally.

► **Proposition 12.** *Suppose in every iteration of Algorithm 1, steps are conducted with respect to randomized gradient operators  $\{g_i(z_t), g_i(w_t^{(i)})\}$  satisfying (14) for some  $\{\bar{w}_t\}$ . Then, for all  $u \in \mathcal{Z}$ ,*

$$\mathbb{E} \left[ \sum_{0 \leq t < T} \langle g(\bar{w}_t), \bar{w}_t - u \rangle \right] \leq \lambda V_{z_0}^r(u).$$

**Proof.** The proof follows identically to that of Proposition 4, where we iterate taking expectations over (4), each time applying the two conditions in (14). ◀

For the rest of this section, we overload  $g_i$  to mean the choices used in (13). This choice is motivated via the following two properties, required by (14).

► **Lemma 13.** *Let  $\bar{w}_t := (x_t + \sum_{i \in [d]} \Delta_t^{(i)}, \nabla f(v_{t+\frac{1}{2}}))$ . Then  $\forall u$ , taking expectations over iteration  $t$ ,*

$$\mathbb{E} \left[ \langle g_i(w_t^{(i)}), w_t^{(i)} - u \rangle \right] = \langle g(\bar{w}_t), \bar{w}_t - u \rangle.$$

**Proof.** Note  $v_{t+\frac{1}{2}}$  is deterministic regardless of the sampled  $i \in [d]$ . Expanding for  $u = (u^x, u^y)$ ,

$$\begin{aligned} \mathbb{E} \left[ \langle g_i(w_t^{(i)}), w_t^{(i)} - u \rangle \right] &= \sum_{i \in [d]} p_i \left( \left\langle \frac{1}{p_i} \nabla_i f(v_{t+\frac{1}{2}}), x_{t+\frac{1}{2}}^{(i)} - u^x \right\rangle \right. \\ &\quad \left. + \left\langle v_{t+\frac{1}{2}} - \left( x_t + \frac{1}{p_i} \Delta_t^{(i)} \right), y_{t+\frac{1}{2}} - u^y \right\rangle \right) \\ &= \langle g(\bar{w}_t), \bar{w}_t - u \rangle. \end{aligned}$$

Here, we used the fact that  $\nabla_i f(v_{t+\frac{1}{2}})$  is 1-sparse. ◀

► **Lemma 14** (Expected relative Lipschitzness). *Let  $\lambda = 1 + S_{1/2}/\sqrt{\mu}$ , where  $S_{1/2} := \sum_{i \in [d]} \sqrt{L_i}$ . Then, for the iterates (13) with  $p_i = \sqrt{L_i}/S_{1/2}$ , taking expectations over iteration  $t$ ,*

$$\mathbb{E} \left[ \langle g_i(w_t^{(i)}) - g_i(z_t), w_t^{(i)} - z_{t+1}^{(i)} \rangle \right] \leq \lambda \mathbb{E} \left[ V_{z_t}^r(w_t^{(i)}) + V_{w_t^{(i)}}^r(z_{t+1}^{(i)}) \right].$$

**Proof.** Equivalently, we wish to show that

$$\mathbb{E} \left[ \left\langle g_i(w_t^{(i)}) - g_i(z_t), w_t^{(i)} - z_{t+1}^{(i)} \right\rangle \right] \leq \left( 1 + \frac{S_{1/2}}{\sqrt{\mu}} \right) \mathbb{E} \left[ V_{z_t}^r(w_t^{(i)}) + V_{w_t^{(i)}}^r(z_{t+1}^{(i)}) \right].$$

The proof is patterned from Lemma 5. By direct calculation, the left hand side is

$$\begin{aligned} & \mathbb{E} \left[ \left\langle g_i(w_t^{(i)}) - g_i(z_t), w_t^{(i)} - z_{t+1}^{(i)} \right\rangle \right] \\ &= \sum_{i \in [d]} p_i \left( \frac{1}{p_i} \left\langle \nabla_i f(v_{t+\frac{1}{2}}) - \nabla_i f(v_t), x_{t+\frac{1}{2}}^{(i)} - x_{t+1}^{(i)} \right\rangle \right. \\ & \quad \left. + \frac{1}{p_i} \left\langle x_t - x_{t+\frac{1}{2}}^{(i)}, \nabla_i f(v_{t+\frac{1}{2}}) - \nabla_i f(v_{t+1}^{(i)}) \right\rangle \right) \\ & \quad + \sum_{i \in [d]} p_i \left\langle v_{t+\frac{1}{2}} - v_t, \nabla f(v_{t+\frac{1}{2}}) - \nabla f(v_{t+1}^{(i)}) \right\rangle. \end{aligned} \tag{15}$$

We first bound the second and third lines of (15):

$$\begin{aligned} & \frac{1}{p_i} \left( \left\langle \nabla_i f(v_{t+\frac{1}{2}}) - \nabla_i f(v_t), x_{t+\frac{1}{2}}^{(i)} - x_{t+1}^{(i)} \right\rangle + \left\langle x_t - x_{t+\frac{1}{2}}^{(i)}, \nabla_i f(v_{t+\frac{1}{2}}) - \nabla_i f(v_{t+1}^{(i)}) \right\rangle \right) \\ & \leq \frac{S_{1/2}}{\sqrt{\mu}} \left( \frac{\mu}{2} \left\| x_{t+\frac{1}{2}}^{(i)} - x_{t+1}^{(i)} \right\|_2^2 + \frac{1}{2L_i} \left\| \nabla_i f(v_{t+\frac{1}{2}}) - \nabla_i f(v_{t+1}^{(i)}) \right\|_2^2 \right. \\ & \quad \left. + \frac{\mu}{2} \left\| x_t - x_{t+\frac{1}{2}}^{(i)} \right\|_2^2 + \frac{1}{2L_i} \left\| \nabla_i f(v_{t+\frac{1}{2}}) - \nabla_i f(v_t) \right\|_2^2 \right) \\ & \leq \frac{S_{1/2}}{\sqrt{\mu}} \left( V_{z_t}^r(w_t^{(i)}) + V_{w_t^{(i)}}^r(z_{t+1}^{(i)}) \right). \end{aligned} \tag{16}$$

The first inequality used the definition  $p_i = \sqrt{L_i/S_{1/2}}$  and Cauchy-Schwarz, and the second used strong convexity and Lemma 13 of the full version of the paper. Next, we bound the fourth line of (15):

$$\begin{aligned} & \left\langle v_{t+\frac{1}{2}} - v_t, \nabla f(v_{t+\frac{1}{2}}) - \mathbb{E} \left[ \nabla f(v_{t+1}^{(i)}) \right] \right\rangle \\ & \leq V_{\nabla f(v_t)}^{f*} \left( \nabla f(v_{t+\frac{1}{2}}) \right) + V_{\nabla f(v_{t+\frac{1}{2}})}^{f*} \left( \mathbb{E} \left[ \nabla f(v_{t+1}^{(i)}) \right] \right) \\ & \leq V_{\nabla f(v_t)}^{f*} \left( \nabla f(v_{t+\frac{1}{2}}) \right) + \mathbb{E} \left[ V_{\nabla f(v_{t+\frac{1}{2}})}^{f*} \left( \nabla f(v_{t+1}^{(i)}) \right) \right] \\ & \leq \mathbb{E} \left[ V_{z_t}^r(w_t^{(i)}) + V_{w_t^{(i)}}^r(z_{t+1}^{(i)}) \right]. \end{aligned}$$

The first inequality is (7), the second is convexity of Bregman divergence, and the third used nonnegativity of  $\frac{\mu}{2} \|\cdot\|_2^2$ . Combining with an expectation over (16) yields the claim.  $\blacktriangleleft$

Crucially, our proof of these results uses the fact that our randomized gradient estimators are 1-sparse in the  $x$  component, and the fact that we “shared randomness” in the definition of the gradient estimators. Moreover, our iterates are efficiently implementable, under the “generalized partial derivative oracle” of prior work [20, 5, 29], which computes  $\nabla_i f(ax + by)$  for  $x, y \in \mathbb{R}^d$  and  $a, b \in \mathbb{R}$ . In many settings of interest, these oracles can be implemented with a dimension-independent runtime; we defer a discussion to previous references.

► **Lemma 15** (Iterate maintenance). *We can implement each iteration of Algorithm 3 using two generalized partial derivative oracle queries and constant additional work.*



We defer a formal statement to the full version of the paper. Combining Lemma 13 and Lemma 14, (14) is satisfied with  $\lambda = 1 + S_{1/2}/\sqrt{\mu}$ . Finally, all of these pieces directly imply the following, via the proof of Theorem 7 and iterating expectations. We give our full method as Algorithm 3.

► **Theorem 16** (Coordinate acceleration). *Algorithm 3 produces an  $\epsilon$ -approximate minimizer of  $f$  in*

$$O\left(\sum_{i \in [d]} \sqrt{\frac{L_i}{\mu}} \log\left(\frac{f(x_0) - f(x^*)}{\epsilon}\right)\right) \text{ iterations in expectation,}$$

with iteration complexity given by Lemma 15.

**Proof.** This follows from the proof of Theorem 7, using Proposition 12 in place of Proposition 4. ◀

■ **Algorithm 3** EG-COORD-ACCEL( $x_0, \epsilon$ ): Extragradient accelerated coordinate minimization.

**Input:**  $x_0 \in \mathbb{R}^d$ ,  $f$   $\{L_i\}_{i \in [d]}$ -coordinate smooth and  $\mu$ -s.c. in  $\|\cdot\|_2$ , and  $\epsilon_0 \geq f(x_0) - f(x^*)$

$$\lambda \leftarrow 1 + \sum_{i \in [d]} \sqrt{L_i/\mu}, T \leftarrow 4\lceil\lambda\rceil, K \leftarrow \lceil\log_2 \frac{\epsilon_0}{\epsilon}\rceil, \mathbf{A} \leftarrow \begin{pmatrix} 1 & \frac{1}{\kappa} - \frac{1}{\kappa^2} \\ 0 & 1 - \frac{1}{\kappa} + \frac{1}{\kappa^2} \end{pmatrix}$$

$p_0 \leftarrow x_0, q_0 \leftarrow x_0, \mathbf{B}_0 \leftarrow \mathbf{I}_{2 \times 2}$

**for**  $0 \leq k < K$  **do**

    Sample  $\tau$  uniformly in  $[0, T - 1]$

**for**  $0 \leq t < \tau$  **do**

        Sample  $i \propto \sqrt{L_i}$

        Compute  $\nabla_i f(v_t), \nabla_i f((1 - \lambda^{-1})v_t + \lambda^{-1}x_t)$  via generalized partial derivative oracle

$$s_t \leftarrow \left( \frac{1}{\mu\lambda p_i} \nabla_i f((1 - \lambda^{-1})v_t + \lambda^{-1}x_t) \quad \frac{1}{\mu\lambda^2 p_i^2} \nabla_i f(v_t) \right)$$

$$\mathbf{B}_{t+1} \leftarrow \mathbf{B}_t \mathbf{A}, (p_{t+1} \quad q_{t+1}) \leftarrow (p_t \quad q_t) - s_t \mathbf{B}_{t+1}^{-1}$$

**end for**

$$\mathbf{B}_0 \leftarrow \begin{pmatrix} [\mathbf{B}_\tau]_{12} & [\mathbf{B}_\tau]_{12} \\ [\mathbf{B}_\tau]_{22} & [\mathbf{B}_\tau]_{22} \end{pmatrix}, p_0 \leftarrow p_\tau, q_0 \leftarrow q_\tau$$

**end for**

**return**  $[\mathbf{B}_\tau]_{12} p_\tau + [\mathbf{B}_\tau]_{22} q_\tau$

## 7 Discussion

We give a general condition for extragradient algorithms to converge at a  $O(T^{-1})$  rate. In turn, we show that this condition (coupled with additional tools such as locality, randomization, or strong monotonicity) yields a recipe for tighter convergence guarantees in structured instances. While our condition applies generally, we find it interesting to broaden the types of instances where it obtains improved runtimes by formulating appropriate VI problems. For example, can we recover acceleration in settings such as finite-sum convex optimization (i.e. for stochastic gradient methods) [3] or composite optimization [7]? Moreover, we are interested in the interplay between (tighter analyses of) extragradient algorithms with other algorithmic frameworks. For example, is there a way to interpolate between our minimax algorithm and the momentum-based framework of [21] to obtain tight runtimes for minimax optimization? Ultimately, our hope is that our methods serve as an important stepping stone towards developing the toolkit for solving e.g. convex-concave games and variational inequalities in general.

## References

- 1 Jacob D. Abernethy, Kevin A. Lai, Kfir Y. Levy, and Jun-Kun Wang. Faster rates for convex-concave games. In *Conference On Learning Theory, COLT 2018, Stockholm, Sweden, 6-9 July 2018.*, pages 1595–1625, 2018.
- 2 Jacob D. Abernethy, Kevin A. Lai, and Andre Wibisono. Last-iterate convergence rates for min-max optimization. *CoRR*, abs/1906.02027, 2019. [arXiv:1906.02027](#).
- 3 Zeyuan Allen Zhu. Katyusha: the first direct acceleration of stochastic gradient methods. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 1200–1205, 2017.
- 4 Zeyuan Allen Zhu and Elad Hazan. Optimal black-box reductions between optimization objectives. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 1606–1614, 2016.
- 5 Zeyuan Allen Zhu, Zheng Qu, Peter Richtárik, and Yang Yuan. Even faster accelerated coordinate descent using non-uniform sampling. In *Proceedings of the 33rd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, pages 1110–1119, 2016.
- 6 Heinz H. Bauschke, Jérôme Bolte, and Marc Teboulle. A descent lemma beyond lipschitz gradient continuity: First-order methods revisited and applications. *Math. Oper. Res.*, 42(2):330–348, 2017.
- 7 Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Imaging Sciences*, 2(1):183–202, 2009.
- 8 Digvijay Boob, Saurabh Sawlani, and Di Wang. Faster width-dependent algorithm for mixed packing and covering lps. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 15253–15262, 2019.
- 9 Sébastien Bubeck. Convex optimization: Algorithms and complexity. *Foundations and Trends in Machine Learning*, 8(3-4):231–357, 2015.
- 10 Yair Carmon, Yujia Jin, Aaron Sidford, and Kevin Tian. Variance reduction for matrix games. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 11377–11388, 2019.
- 11 Yair Carmon, Yujia Jin, Aaron Sidford, and Kevin Tian. Coordinate methods for matrix games. In *61st Annual IEEE Symposium on Foundations of Computer Science, FOCS 2020*, 2020.
- 12 Tatjana Chavdarova, Gauthier Gidel, François Fleuret, and Simon Lacoste-Julien. Reducing noise in GAN training with variance reduced extragradient. *CoRR*, abs/1904.08598, 2019. [arXiv:1904.08598](#).
- 13 Jelena Diakonikolas and Lorenzo Orecchia. Accelerated extra-gradient descent: A novel accelerated first-order method. In *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, pages 23:1–23:19, 2018.
- 14 Radu-Alexandru Dragomir, Adrien Taylor, Alexandre d’Aspremont, and Jérôme Bolte. Optimal complexity and certification of bregman first-order methods. *CoRR*, abs/1911.08510, 2019. [arXiv:1911.08510](#).
- 15 Filip Hanzely, Peter Richtárik, and Lin Xiao. Accelerated bregman proximal gradient methods for relatively smooth convex optimization. *CoRR*, abs/1808.03045, 2018. [arXiv:1808.03045](#).
- 16 Yu-Guan Hsieh, Franck Iutzeler, Jérôme Malick, and Panayotis Mertikopoulos. On the convergence of single-call stochastic extra-gradient methods. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 6936–6946, 2019.

- 17 Arun Jambulapati, Aaron Sidford, and Kevin Tian. A direct  $\tilde{O}(1/\epsilon)$  iteration parallel algorithm for optimal transport. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 11355–11366, 2019.
- 18 Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pages 315–323, 2013.
- 19 Anatoli Juditsky, Arkadi Nemirovski, and Claire Tauvel. Solving variational inequalities with stochastic mirror-prox algorithm. *Stochastic Systems*, 1(1):17–58, 2011.
- 20 Yin Tat Lee and Aaron Sidford. Efficient accelerated coordinate descent methods and faster algorithms for solving linear systems. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 147–156, 2013.
- 21 Tianyi Lin, Chi Jin, and Michael I. Jordan. Near-optimal algorithms for minimax optimization. *CoRR*, abs/2002.02417, 2020. [arXiv:2002.02417](https://arxiv.org/abs/2002.02417).
- 22 Haihao Lu. “relative-continuity” for non-lipschitz non-smooth convex optimization using stochastic (or deterministic) mirror descent. *INFORMS Journal on Optimization*, pages 288–303, 2019.
- 23 Haihao Lu, Robert M. Freund, and Yurii E. Nesterov. Relatively smooth convex optimization by first-order methods, and applications. *SIAM J. Optim.*, 28(1):333–354, 2018.
- 24 Panayotis Mertikopoulos, Bruno Lecouat, Houssam Zenati, Chuan-Sheng Foo, Vijay Chandrasekhar, and Georgios Piliouras. Optimistic mirror descent in saddle-point problems: Going the extra (gradient) mile. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019.
- 25 A. Nemirovski and D.B. Yudin. *Problem Complexity and Method Efficiency in Optimization*. Wiley, 1983.
- 26 Arkadi Nemirovski. Prox-method with rate of convergence  $o(1/t)$  for variational inequalities with lipschitz continuous monotone operators and smooth convex-concave saddle point problems. *SIAM Journal on Optimization*, 15(1):229–251, 2004.
- 27 Yurii Nesterov. A method for solving a convex programming problem with convergence rate  $o(1/k^2)$ . *Doklady AN SSSR*, 269:543–547, 1983.
- 28 Yurii Nesterov. Dual extrapolation and its applications to solving variational inequalities and related problems. *Math. Program.*, 109(2-3):319–344, 2007.
- 29 Yurii Nesterov and Sebastian U. Stich. Efficiency of the accelerated coordinate descent method on structured optimization problems. *SIAM Journal on Optimization*, 27(1):110–123, 2017.
- 30 Yuyuan Ouyang and Yangyang Xu. Lower complexity bounds of first-order methods for convex-concave bilinear saddle-point problems. *Mathematical Programming*, 2019.
- 31 Balamurugan Palaniappan and Francis R. Bach. Stochastic variance reduction methods for saddle-point problems. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 1408–1416, 2016.
- 32 Alexander Rakhlin and Karthik Sridharan. Optimization, learning, and games with predictable sequences. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 3066–3074, 2013.
- 33 Mark W. Schmidt, Nicolas Le Roux, and Francis R. Bach. Minimizing finite sums with the stochastic average gradient. *Math. Program.*, 162(1-2):83–112, 2017.
- 34 Jonah Sherman. Area-convexity,  $l_\infty$  regularization, and undirected multicommodity flow. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 452–460, 2017.

- 35    Aaron Sidford and Kevin Tian. Coordinate methods for accelerating  $\ell_\infty$  regression and faster approximate maximum flow. In *59th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2018, 7-9 October, 2018, Paris, France, 2018*.
- 36    Fedor Stonyakina, Alexander Tyurin, Alexander Gasnikov, Pavel Dvurechensky, Artem Agafonov, Darina Dvinskikh, Dmitry Pasechnyuk, Sergei Artamonov, and Victorya Piskunova. Inexact relative smoothness and strong convexity for optimization and variational inequalities by inexact model. *CoRR*, abs/2001.09013, 2020. [arXiv:2001.09013](#).
- 37    Jun-Kun Wang and Jacob D. Abernethy. Acceleration through optimistic no-regret dynamics. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada.*, pages 3828–3838, 2018.
- 38    Junyu Zhang, Minyi Hong, and Shuzhong Zhang. On lower iteration complexity bounds for the saddle point problems. *CoRR*, abs/1912.07481, 2019. [arXiv:1912.07481](#).

# Training (Overparametrized) Neural Networks in Near-Linear Time

**Jan van den Brand**

KTH Royal Institute of Technology, Stockholm, Sweden  
janvdb@kth.se

**Binghui Peng**

Columbia University, New York, NY, USA  
bp2601@columbia.edu

**Zhao Song**

Princeton University and Institute for Advanced Study, NJ, USA  
zhaos@ias.edu

**Omri Weinstein**

Columbia University, New York, NY, USA  
omri@cs.columbia.edu

---

## Abstract

The slow convergence rate and pathological curvature issues of first-order gradient methods for training deep neural networks, initiated an ongoing effort for developing faster *second-order* optimization algorithms beyond SGD, without compromising the generalization error. Despite their remarkable convergence rate (*independent* of the training batch size  $n$ ), second-order algorithms incur a daunting slowdown in the *cost per iteration* (inverting the Hessian matrix of the loss function), which renders them impractical. Very recently, this computational overhead was mitigated by the works of [79, 23], yielding an  $O(mn^2)$ -time second-order algorithm for training two-layer overparametrized neural networks of polynomial width  $m$ .

We show how to speed up the algorithm of [23], achieving an  $\tilde{O}(mn)$ -time backpropagation algorithm for training (mildly overparametrized) ReLU networks, which is near-linear in the dimension  $(mn)$  of the full gradient (Jacobian) matrix. The centerpiece of our algorithm is to reformulate the Gauss-Newton iteration as an  $\ell_2$ -regression problem, and then use a Fast-JL type dimension reduction to *precondition* the underlying Gram matrix in time independent of  $M$ , allowing to find a sufficiently good approximate solution via *first-order* conjugate gradient. Our result provides a proof-of-concept that advanced machinery from randomized linear algebra – which led to recent breakthroughs in *convex optimization* (ERM, LPs, Regression) – can be carried over to the realm of deep learning as well.

**2012 ACM Subject Classification** Theory of computation → Nonconvex optimization

**Keywords and phrases** Deep learning theory, Nonconvex optimization

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.63

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/2006.11648>.

**Funding** *Jan van den Brand*: This project has received funding from the European Research Council (ERC) under the European Unions Horizon 2020 research and innovation program under grant agreement No 715672. Partially supported by the Google PhD Fellowship Program.

*Binghui Peng*: Research supported by NSF IIS-1838154, NSF CCF-1703925 and NSF CCF-1763970.

*Zhao Song*: Research supported by Special Year on Optimization, Statistics, and Theoretical Machine Learning (being led by Sanjeev Arora) at Institute for Advanced Study.

*Omri Weinstein*: Research supported by NSF CAREER award CCF-1844887.

**Acknowledgements** The author would like to thank David Woodruff for telling us the tensor trick for computing kernel matrices and helping us improve the presentation of the paper.



© Jan van den Brand, Binghui Peng, Zhao Song, and Omri Weinstein;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 63; pp. 63:1–63:15



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

## 1 Introduction

Understanding the dynamics of gradient-based optimization of deep neural networks has been a central focal point of theoretical machine learning in recent years [49, 81, 80, 48, 31, 4, 5, 3, 13, 58, 9, 67, 27, 39, 22]. This line of work led to a remarkable rigorous understanding of the generalization, robustness and convergence rate of *first-order* (SGD-based) algorithms, which are the standard choice for training DNNs. By contrast, the *computational complexity* of implementing gradient-based training algorithms (e.g., backpropagation) in such non-convex landscape is less understood, and gained traction only recently due to the overwhelming size of training data and complexity of network design [55, 32, 51, 23, 79].

The widespread use first-order methods such as (stochastic) gradient descent in training DNNs is explained, to a large extent, by its computational efficiency – recalculating the gradient of the loss function at each iteration is simple and cheap (linear in the dimension of the full gradient), let alone with the advent of minibatch random sampling [37, 23]. Nevertheless, first-order methods have a slow rate of convergence in non-convex settings (typically  $\Omega(\text{poly}(n) \log(1/\epsilon))$  for overparametrized networks, see e.g., [79]) for reducing the training error below  $\epsilon$ , and it is increasingly clear that SGD-based algorithms are becoming a real bottleneck for many practical purposes. This drawback initiated a substantial effort for developing fast training methods beyond SGD, aiming to improve its convergence rate without compromising the generalization error [16, 53, 55, 32, 44, 59, 23, 79].

Second-order gradient algorithms (which employ information about the Hessian of the loss function), pose an intriguing computational tradeoff in this context: On one hand, they are known to converge extremely fast, at a rate *independent* of the input size (i.e., only  $O(\log 1/\epsilon)$  iterations [79]), and offer a qualitative advantage in overcoming pathological curvature issues that arise in first-order methods, by exploiting the local geometry of the loss function. This feature implies another practical advantage of second order methods, namely, that they do not require tuning the learning rate [23, 79]. On the other hand, second-order methods have a prohibitive *cost per iteration*, as they involve *inverting* a dynamically-changing dense Hessian matrix. This drawback explains the scarcity of second order methods in *large scale non-convex* optimization, in contrast to its popularity in the convex setting.

The recent works of [23, 79] addressed the computational bottleneck of second-order algorithms in optimizing deep neural nets, and presented a training algorithm for overparametrized neural networks with smooth (resp. ReLU) activations, whose running time is  $O(mn^2)$ , where  $m$  is the width of the neural network, and  $n$  is the size of the training data in  $\mathbb{R}^d$ . The two algorithms, which achieve essentially the same running time, are based on the classic Gauss-Newton algorithm (resp. “Natural gradient” algorithm) combined with the recent introduction of *Neural Tangent Kernels* (NTK) [38]. The NTK formulation utilizes a local-linearization of the loss function for overparametrized neural networks, which reduces the optimization problem of DNNs to that of a *kernel regression* problem: The main insight is that when the network is *overparametrized*, i.e., sufficiently wide  $m \gtrsim n^4$  ([67]), the neural network becomes locally convex and smooth, hence the problem is equivalent to a kernel regression problem with respect to the NTK function [38], and therefore solving the latter via (S)GD is guaranteed to converge to a global minimum. The training algorithm of [23] draws upon this equivalence, by designing a second-order variation of the Gauss-Newton algorithm (termed “Gram-Gauss-Newton”), yielding the aforementioned runtime for *smooth activation functions*.

**Single vs. Multilayer Network Training.** Following [23, 79], we focus on two-layer (i.e., single hidden-layer) neural networks. While our algorithm extends to the multilayer case (with a slight compromise on the width dependence), we argue that, as far as training time,

the two-layer case is not only the common case, but in fact the *only* interesting case for constant training error: Indeed, in the multilayer case ( $L \geq 2$ ), we claim that the mere cost of *feed-forward* computation of the network's output is already  $\Omega_\epsilon(m^2 n L)$ . Indeed, the total number of parameters of  $L$ -layer networks is  $M = (L - 1)m^2 + md$ , and as such, feed-forward computation requires, at the very least, computing a single product of  $m \times m$  (dense) matrices  $W$  with a  $m \times 1$  vector for each training data, which already costs  $m^2 n$  time:

$$\hat{y}_i = a^\top \sigma_L \left( \underbrace{W_L}_{m \times m} \sigma_{L-1} \left( \underbrace{W_{L-1}}_{m \times m} \dots \sigma_1 \left( \underbrace{W_1}_{m \times d} x_i \right) \right) \right).$$

Therefore, sublinear-time techniques (as we present) appear futile in the case of multilayer overparametrized networks, where it is possible to achieve linear time (in  $M$ ) using essentially direct (lossless) computation (see next subsection). It may still be possible to use sublinear algorithms to improve the running time to  $O(m^2 n L + \text{poly}(n))$ , though in for overparametrized DNNs this seems a minor saving.

## 1.1 Our Result

Our main result is a quadratic speedup to the algorithm of [23], yielding an *essentially optimal* training algorithm for overparametrized two-layer neural networks. Moreover, in contrast to [23], our algorithm applies to the more complex and realistic case of *ReLU* activation functions. Our main result is shown below (For a more comprehensive comparison, see Table 1 below and references therein).

► **Theorem 1.** *Suppose the width of a two layer ReLU neural network satisfies*

$$m = \Omega(\max\{\lambda^{-4} n^4, \lambda^{-2} n^2 d \log(n/\delta)\}),$$

where  $\lambda > 0$  denotes the minimum eigenvalue of the Gram matrix (see Eq. (5) below),  $n$  is the number of training data,  $d$  is the input dimension. Then with probability  $1 - \delta$  over the random initialization of neural network and the randomness of the training algorithm, our algorithm achieves

$$\|f_{t+1} - y\|_2 \leq \frac{1}{2} \|f_t - y\|_2.$$

The computational cost of each iteration is  $\tilde{O}(mnd + n^3)$ , and the running time for reducing the training loss to  $\epsilon$  is  $\tilde{O}((mnd + n^3) \log(1/\epsilon))$ . Using fast matrix-multiplication, the total running time can be further reduced to  $\tilde{O}((mnd + n^\omega) \log(1/\epsilon))$ .<sup>1</sup>

► **Remark 2.** We stress that that our algorithm runs in (near) linear time even for networks with width  $m \gtrsim n^2$  and in fact, under the common belief that  $\omega = 2$ , this is true so long as  $m \gtrsim n$  (!). This means that the bottleneck for linear-time training of *small-width* DNNs is *not computational*, but rather *analytic*: The overparametrization requirements ( $m \gtrsim n^4$ ) in Theorem 1 stems from current-best analysis of the convergence guarantees of (S)GD-based training of ReLU networks, and any improvement on these bounds would directly yield linear-time training for thinner networks using our algorithm.

<sup>1</sup> Here,  $\omega < 2.373$  denotes the fast matrix-multiplication (FMM) constant for multiplying two  $n \times n$  matrices [73, 46].



■ **Table 1** Summary of state-of-art algorithms for training two-layer neural networks.  $n$  denotes the training batch size (number of input data points in  $\mathbb{R}^d$ ) and  $\epsilon$  denote the desired accuracy of the training loss. For simplicity, here we assume  $d = O(1)$  and omit  $\text{poly}(\log n, 1/\lambda)$  terms. The result of [23] applies only to smooth activation gates and not to ReLU networks. Comparison to SGD algorithms is omitted from this table since they require a much stronger assumption on the width  $m$  for convergence, and have slower convergence rate than GD [48, 4, 5].

Ref.	Method	#Iters	Cost/iter	Width	ReLU?
[31]	Gradient descent	$O(n^2 \log(1/\epsilon))$	$O(mn)$	$\Omega(n^6)$	Yes
[67]	Gradient descent	$O(n^2 \log(1/\epsilon))$	$O(mn)$	$\Omega(n^4)$	Yes
[77]	Adaptive gradient descent	$O(n \log(1/\epsilon))$	$O(mn)$	$\Omega(n^6)$	Yes
[23]	Gram-Gaussian-Newton (GGN)	$O(\log \log(1/\epsilon))$	$O(mn^2)$	$\Omega(n^4)$	No
[23]	Batch-GGN	$O(n^2 \log(1/\epsilon))$	$O(m)$	$\Omega(n^{18})$	No
[79]	Natural gradient descent	$O(\log(1/\epsilon))$	$O(mn^2)$	$\Omega(n^4)$	Yes
<b>Ours</b>		$O(\log(1/\epsilon))$	$O(mn)$	$\Omega(n^4)$	Yes

**Techniques.** The majority of ML optimization literature on overparametrized network training is dedicated to understanding and minimizing the *number of iterations* of the training process [79, 23] as opposed to the *cost per iteration*, which is the focus of our paper. Our work shows that it is possible to harness the toolbox of *randomized linear algebra* – which was heavily used in the past decade to reduce the cost of *convex optimization* tasks – in the nonconvex setting of deep learning as well. A key ingredient in our algorithm is *linear sketching*, where the main idea is to carefully *compress* a linear system underlying an optimization problem, in a way that preserves a good enough solution to the problem yet can be solved much faster in lower dimension. This is the essence of the celebrated *Sketch-and-Solve* (S&S) paradigm [24]. As we explain below, our main *departure* from the classic S&S framework (e.g., [59]) is that we cannot afford to directly solve the underlying compressed regression problem (as this approach turns out to be prohibitively slow for our application). Instead, we use sketching (or sampling) to facilitate *fast preconditioning* of linear systems (in the spirit of [68, 43, 62, 74]), which in turn enables to solve the compressed regression problem to very high accuracy via first-order *conjugate* gradient descent. This approach essentially *decouples* the sketching error from the final precision error of the Gauss-Newton step, enabling a much smaller sketch size. We believe this (somewhat unconventional) approach to non-convex optimization is the most enduring message of our work.

## 1.2 Related Work

**Second-order methods in non-convex optimization.** Despite the prevalence of first order methods in deep learning applications, there is a vast body of ongoing work [18, 17, 55, 35, 36, 23, 79] aiming to design more scalable second-order algorithms that overcome the limitations of (S)GD for optimizing deep models. Grosse and Martens [55, 35] designed the K-FAC method, where the idea is to use Kronecker-factors to approximate the Fisher information matrix, combined with natural gradient descent. This approach has been further explored and extended by [78, 34, 54]. Gupta et al. [36] designed the “Shampoo method”, based on the idea of *structure-aware preconditioning*. Anil et al. [7] further validate the practical performance of Shampoo and incorporated it into hardware. However, despite sporadic empirical evidence of such second-order methods (e.g., K-FAC and Shampoo), these methods generally lack a provable theoretical guarantee on the performance when applied to deep neural networks. Furthermore, in the overparametrized setting, their cost per-iteration in general is at least  $\Omega(mn^2)$ .

We remark that in the *convex* setting, theoretical guarantees for large-scale second-order algorithms have been established (e.g., [1, 59, 56, 21]), but such rigorous analysis in non-convex setting was only recently proposed ([23, 79]). Our algorithm bears some similarities to the *NewtonSketch* algorithm of [59], which also incorporates sketching into second order Newton methods. A key difference, however, is that the algorithm of [59] works only for convex problems, and requires access to  $(\nabla^2 f(x))^{1/2}$  (i.e., the square-root of the Hessian). Most importantly, though, [59] use the standard (black-box) Sketch-and-Solve paradigm to reduce the computational cost, while this approach incurs large computation overhead in our non-convex setting. By contrast, we use sketching as a subroutine for fast preconditioning. As a by-product, in the full version of this paper we show how to apply our techniques to give a substantial improvement over [59] in the *convex* setting.

The aforementioned works of [79] and [23] are most similar in spirit to ours. Zhang et al. [79] analyzed the convergence rate of Natural gradient descent algorithms for two-layer (overparametrized) neural networks, and showed that the number of iterations is *independent* of the training data size  $n$  (essentially  $\log(1/\epsilon)$ ). They also demonstrate similar results for the convergence rate of K-FAC in the overparametrized regime, albeit with larger requirement on the width  $m$ . Another downside of K-FAC is the high cost per iteration ( $\sim mn^2$ ). Cai et al. [23] analyzed the convergence rate of the so-called Gram-Gauss-Newton algorithm for training two-layer (overparametrized) neural network with *smooth* activation gates. They proved a quadratic (i.e., doubly-logarithmic) convergence rate in this setting ( $\log(\log(1/\epsilon))$ ) albeit with  $O(mn^2)$  cost per iteration. It is noteworthy that this quadratic convergence rate analysis does not readily extend to the more complex and realistic setting of ReLU activation gates, which is the focus of our work. [23] also prove bounds on the convergence of “batch GGN”, showing that it is possible to reduce the cost-per-iteration to  $m$ , at the price of  $O(n^2 \log(1/\epsilon))$  iterations, for very heavily overparametrized DNNs (currently  $m = \Omega(n^{18})$ ).

**Sketching.** The celebrated “Sketch and Solve” (S&S) paradigm [24] was originally developed to speed up the cost of solving linear regression and low-rank approximation problems. This dimensionality-reduction technique has since then been widely developed and applied to both convex and non-convex numerical linear algebra problems [20, 61, 76, 6, 14, 12, 72, 28, 65, 64, 15], as well as machine-learning applications [10, 11, 50, 75]. The most direct application of the sketch-and-solve technique is overconstrained regression problems, where the input is a linear system  $[A, b] \in \mathbb{R}^{n \times (d+1)}$  with  $n \gg d$ , and we aim to find an (approximate) solution  $\hat{x} \in \mathbb{R}^d$  so as to minimize the residual error  $\|A\hat{x} - b\|_2$ .

In the classic S&S paradigm, the underlying regression solver is treated as a *black box*, and the computational savings comes from applying it on a smaller *compressed* matrix. Since then, sketching (or sampling) has also been used in a non-black-box fashion for speeding-up optimization tasks, e.g., as a subroutine for preconditioning [74, 62, 68, 43] or fast inverse-maintenance in Linear Programming solvers, semi-definite programming, cutting plane methods, and empirical-risk minimization [25, 42, 40, 41, 47].

**Overparametrization in neural networks.** A long and active line of work in recent deep learning literature has focused on obtaining rigorous bounds on the convergence rate of various local-search algorithms for optimizing DNNs [48, 31, 4, 5, 8, 9, 67, 39]. The breakthrough work of Jacob et al. [38] and subsequent developments<sup>2</sup> introduced the notion of *neural tangent kernels* (NTK), implying that for wide enough networks ( $m \gtrsim n^4$ ), (stochastic) gradient descent provably converges to an optimal solution, with generalization error independent of the number of network parameters.

<sup>2</sup> For a complete list of references, we refer the readers to [8, 9].

## 2 Technical Overview

We now provide a streamlined overview of our main result, Theorem 1. As discussed in the introduction, our algorithm extends to multi-layer ReLU networks, though we focus on the two-layer case (one-hidden layer), which is the most interesting case where one can indeed hope for linear training time.

The main, and most expensive step, of the GGN (or natural gradient descent) algorithms [23, 79] is multiplying, in each iteration  $t$ , the *inverse* of the Gram matrix  $G_t := J_t J_t^\top$  with the Jacobian matrix  $J_t \in \mathbb{R}^{n \times m}$ , whose  $i$ th row contains the gradient of the  $m = md$  network gates w.r.t the  $i$ th datapoint  $x_i$  (in our case, under ReLU activation).

Naively computing  $G_t$  would already take  $mdn^2$  time, however, the *tensor product* structure of the Jacobian  $J$  in fact allows to compute  $G_t$  in  $n \cdot \mathcal{T}_{mat}(m, d, n) \ll mn^2$  time, where  $\mathcal{T}_{mat}(m, d, n)$  is the cost of fast rectangular matrix multiplication [73, 46, 33].<sup>3</sup> Since the Gram-Gauss-Newton (GGN) algorithm requires  $O(\log \log 1/\epsilon)$  iterations to converge to an  $\epsilon$ -global minimum of the  $\ell_2$  loss [23], this observation yields an  $O(n \cdot \mathcal{T}_{mat}(m, d, n) \log \log 1/\epsilon)$  total time algorithm for reducing the training loss below  $\epsilon$ . While already nontrivial, this is still far from linear running time ( $\gg mdn$ ).

We show how to carry out each Gauss-Newton iteration in time  $\tilde{O}(mnd + n^3)$ , at the price of slightly compromising the number of iterations to  $O(\log 1/\epsilon)$ , which is inconsequential for the natural regime of constant dimension  $d$  and constant  $\epsilon$ <sup>4</sup>. Our first key step is to reformulate the Gauss-Newton iteration (multiplying  $G_t^{-1}$  by the error vector) as an  *$\ell_2$ -regression problem*:

$$\min_{g_t} \|J_t J_t^\top g_t - (f_t - y)\|_2 \quad (1)$$

where  $(f_t - y)$  is the training error with respect to the network's output and the training labels  $y$ . Since the Gauss-Newton method is robust to small perturbation errors (essentially [71, 70]), our analysis shows that it is sufficient to find an approximate solution  $g'_t$  such that  $J_t^\top g'_t$  satisfies

$$\|J_t J_t^\top g'_t - y\|_2 \leq \gamma \|y\|_2, \quad \text{for } \gamma \approx 1/n. \quad (2)$$

The benefit of this reformulation is that it allows to use *linear sketching* to first compress the linear system, significantly reducing the dimension of the optimization problem and thereby the cost of finding a solution, at the price of a small error in the found solution (this is the essence of the *sketch-and-solve* paradigm [24]). Indeed, a (variation of) the *Fast-JL* sketch [2, 52] guarantees that we can multiply the matrix  $J_t^\top \in \mathbb{R}^{m \times n}$  by a much smaller  $\tilde{O}(n/\delta^2) \times m$  matrix  $S$ , such that (i) the multiplication takes near-linear time  $\tilde{O}(mn)$  time (using the FFT algorithm), and (ii)  $SJ_t^\top$  is a  $\delta$ -spectral approximation of  $J_t^\top$  (i.e.,  $\|J_t S^\top S J_t^\top x\|_2 = (1 \pm \delta) \|G_t x\|_2$  for every  $x$ ). Since both computing and inverting the matrix  $\tilde{G}_t := J_t S^\top S J_t^\top$  takes  $\tilde{O}(n^3/\delta^2)$  time, the overall cost of finding a  $\delta$ -approximate solution to

<sup>3</sup> To see this, observe that the kronecker-product structure of  $J$  (here  $J \in \mathbb{R}^{n \times md}$  can be constructed from an  $n \times m$  matrix and an  $n \times d$  matrix) allows computing  $Jh$  for any  $h \in \mathbb{R}^{md}$  using fast rectangular matrix multiplication in time  $\mathcal{T}_{mat}(m, d, n)$  which is near linear time in the dimension of  $J$  and  $h$  (that is,  $n \times m + n \times d$  for  $J$  and  $md$  for  $h$ ) so long as  $d \leq n^\alpha = n^{0.31}$  [33], hence computing  $G = JJ^\top$  can be done using  $n$  independent invocations of the aforementioned subroutine, yielding  $n \cdot \mathcal{T}_{mat}(m, d, n)$  as claimed.

<sup>4</sup> We also remark that this slowdown in the convergence rate is also a consequence of a direct extension of the analysis in [23] to ReLU activation functions.

the regression problem becomes at most  $\tilde{O}(mn + n^3/\delta^2)$ . Alas, as noted in Equation (2), the approximation error of the found solution must be *polynomially small*  $\gamma \sim 1/n$  in order to guarantee the desired convergence rate (i.e., constant decrease in training error per iteration). This means that we must set  $\delta \sim \gamma \sim 1/n$ , hence the cost of the naïve “sketch-and-solve” algorithm would be at least  $\tilde{O}(n^3/\delta^2) = \tilde{O}(n^5)$ , which is a prohibitively large overhead in both theory and practice (and in particular, no longer yields linear runtime whenever  $m \ll n^4$  which is the current best overparametrization guarantee [67]). Since the  $O(1/\delta^2)$  dependence of the JL embedding is known to be tight in general [45], this means we need to take a more clever approach to solve the regression (1). This is where our algorithm departs from the naïve sketch-and-solve method, and is the heart of our work.

Our key idea is to use dimension reduction – not to directly invert the compressed matrix – but rather to *precondition* it quickly. More precisely, our approach is to use a (conjugate) gradient-descent solver for the regression problem itself, with a fast preconditioning step, ensuring exponentially faster convergence to very high (polynomially small) accuracy. Indeed, conjugate gradient descent is guaranteed to find a  $\gamma$ -approximate solution to a regression problem  $\min_x \|Ax - b\|_2$  in  $O(\sqrt{\kappa} \log(1/\gamma))$  iterations, where  $\kappa(A)$  is the *condition number* of  $A$  (i.e., the ratio of maximum to minimum eigenvalue). Therefore, if we can ensure that  $\kappa(G_t)$  is small, then we can  $\gamma$ -solve the regression problem in  $\sim mn \log(1/\gamma) = \tilde{O}(mn)$  time, since the per-iteration cost of first-order SGD is linear ( $\sim mn$ ).

The crucial advantage of our approach is that it *decouples* the sketching error from the final precision of the regression problem: Unlike the usual “sketch-and-solve” method, where the sketching error  $\delta$  directly affects the overall precision of the solution to (2), here  $\delta$  only affects the *quality of the preconditioner* (i.e., the ratio of max/min singular values of the sketch  $\tilde{G}_t$ ), hence it suffices to take a *constant* sketching error  $\delta = 0.1$  (say), while letting the SGD deal with the final precision (at it has logarithmic dependence on  $\gamma$ ).

Indeed, by setting the sketching error to  $\delta = 0.1$  (say), the resulting matrix  $\tilde{G}_t = J_t S^\top S J_t^\top$  is small enough ( $n \times \tilde{O}(n)$ ) that we can afford running a standard (QR) algorithm to precondition it, at another  $\tilde{O}(n^3)$  cost per iteration. The output of this step is a matrix  $\tilde{G}'_t := \text{Prec}(\tilde{G}_t)$  with a *constant* condition number  $\kappa(\tilde{G}'_t)$  which preserves  $\tilde{G}'_t x \approx_{\ell_2} \tilde{G}_t x$  up to  $(1 \pm \delta)^2$  relative error. At this point, we can run a (conjugate) gradient descent algorithm, which is guaranteed to find a  $\gamma \approx 1/n$  approximate solution to (1) in time  $\tilde{O}((mn \log((1 + \delta)/\gamma) + n^3))$ , as desired.

We remark that, by definition, the preconditioning step (on the JL sketch) does *not* preserve the eigen-spectrum of  $G_t$ , which is in fact necessary to guarantee the fast convergence of the Gauss-Newton iteration. The point is that this preconditioning step is only preformed as a *local subroutine* so as to solve the regression problem, and does *not* affect the convergence rate of the outer loop.

### 3 Preliminaries

#### 3.1 Model and Problem Setup

We denote by  $n$  the number of data points in the training batch, and by  $d$  the data dimension/feature-space (i.e.,  $x_i \in \mathbb{R}^d$ ). We denote by  $m$  the *width* of neural network, and by  $L$  the number of layers and by  $M$  the number of parameters. We assume the data has been normalized, i.e.,  $\|x\|_2 = 1$ . We begin with the two-layer neural network in the following section, and then extend to multilayer networks. Consider a two-layer ReLU activated neural network with  $m$  neurons in the (single) hidden layer:

$$f(W, x, a) = \frac{1}{\sqrt{m}} \sum_{r=1}^m a_r \phi(w_r^\top x),$$

where  $x \in \mathbb{R}^d$  is the input,  $w_1, \dots, w_m \in \mathbb{R}^d$  are weight vectors in the first layer,  $a_1, \dots, a_m \in \mathbb{R}$  are weights in the second layer. For simplicity, we consider  $a \in \{-1, +1\}^m$  is fixed over all the iterations, this is natural in deep learning theory [48, 31, 4, 3, 67]. Recall the ReLU function  $\phi(x) = \max\{x, 0\}$ . Therefore for  $r \in [m]$ , we have

$$\frac{\partial f(W, x, a)}{\partial w_r} = \frac{1}{\sqrt{m}} a_r x \mathbf{1}_{w_r^\top x \geq 0}. \quad (3)$$

Given  $n$  input data points  $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n) \in \mathbb{R}^d \times \mathbb{R}$ . We define the objective function  $\mathcal{L}$  as follows

$$\mathcal{L}(W) = \frac{1}{2} \sum_{i=1}^n (y_i - f(W, x_i, a))^2.$$

We can compute the gradient of  $\mathcal{L}$  in terms of  $w_r$

$$\frac{\partial \mathcal{L}(W)}{\partial w_r} = \frac{1}{\sqrt{m}} \sum_{i=1}^n (f(W, x_i, a) - y_i) a_r x_i \mathbf{1}_{w_r^\top x_i \geq 0}. \quad (4)$$

We define the prediction function  $f_t : \mathbb{R}^{d \times n} \rightarrow \mathbb{R}^n$  at time  $t$  as follow

$$f_t = \begin{bmatrix} \frac{1}{\sqrt{m}} \sum_{r=1}^m a_r \cdot \phi(\langle w_r(t), x_1 \rangle) \\ \frac{1}{\sqrt{m}} \sum_{r=1}^m a_r \cdot \phi(\langle w_r(t), x_2 \rangle) \\ \vdots \\ \frac{1}{\sqrt{m}} \sum_{r=1}^m a_r \cdot \phi(\langle w_r(t), x_n \rangle) \end{bmatrix}$$

where  $W_t = [w_1(t)^\top, w_2(t)^\top, \dots, w_m(t)^\top]^\top \in \mathbb{R}^{md}$  and  $X = [x_1, x_2, \dots, x_n] \in \mathbb{R}^{d \times n}$ .

For each time  $t$ , the Jacobian matrix  $J \in \mathbb{R}^{n \times md}$  is defined via the following formulation:

$$J_t = \frac{1}{\sqrt{m}} \begin{bmatrix} a_1 x_1^\top \mathbf{1}_{\langle w_1(t), x_1 \rangle \geq 0} & a_2 x_1^\top \mathbf{1}_{\langle w_2(t), x_1 \rangle \geq 0} & \cdots & a_m x_1^\top \mathbf{1}_{\langle w_m(t), x_1 \rangle \geq 0} \\ a_1 x_2^\top \mathbf{1}_{\langle w_1(t), x_2 \rangle \geq 0} & a_2 x_2^\top \mathbf{1}_{\langle w_2(t), x_2 \rangle \geq 0} & \cdots & a_m x_2^\top \mathbf{1}_{\langle w_m(t), x_2 \rangle \geq 0} \\ \vdots & \vdots & \ddots & \vdots \\ a_1 x_n^\top \mathbf{1}_{\langle w_1(t), x_n \rangle \geq 0} & a_2 x_n^\top \mathbf{1}_{\langle w_2(t), x_n \rangle \geq 0} & \cdots & a_m x_n^\top \mathbf{1}_{\langle w_m(t), x_n \rangle \geq 0} \end{bmatrix}.$$

The Gram matrix  $G_t$  is defined as  $G_t = J_t J_t^\top$ , whose  $(i, j)$ -th entry is  $\left\langle \frac{f(W_t, x_i)}{\partial W}, \frac{f(W_t, x_j)}{\partial W} \right\rangle$ . The crucial observation of [38, 31] is that the asymptotic of the Gram matrix equals a positive semidefinite kernel matrix  $K \in \mathbb{R}^{n \times n}$ , where

$$K(x_i, x_j) = \mathbb{E}_{w \in \mathcal{N}(0, 1)} [x_i^\top x_j \mathbf{1}_{\langle w, x_i \rangle \geq 0, \langle w, x_j \rangle \geq 0}]. \quad (5)$$

► **Assumption 3.** We assume the least eigenvalue  $\lambda$  of the kernel matrix  $K$  defined in Eq. (5) satisfies  $\lambda > 0$ .

### 3.2 Subspace embedding

Subspace embedding was first introduced by Sarlós [63], it has been extensively used in numerical linear algebra field over the last decade [24, 57, 19, 66]. For a more detailed survey, we refer the readers to [74]. The formal definition is:

► **Definition 4** (Approximate subspace embedding, ASE [63]). A  $(1 \pm \epsilon)$   $\ell_2$ -subspace embedding for the column space of an  $N \times k$  matrix  $A$  is a matrix  $S$  for which for all  $x \in \mathbb{R}^k$ ,  $\|SAx\|_2^2 = (1 \pm \epsilon)\|Ax\|_2^2$ . Equivalently,  $\|I - U^\top S^\top S U\|_2 \leq \epsilon$ , where  $U$  is an orthonormal basis for the column space of  $A$ .

Combining Fast-JL sketching matrix [2, 30, 69, 29, 52, 60] with a classical  $\epsilon$ -net argument [74] gives subspace embedding,

► **Lemma 5** (Fast subspace embedding [52, 74]). *Given a matrix  $A \in \mathbb{R}^{N \times k}$  with  $N = \text{poly}(k)$ , then we can compute a  $S \in \mathbb{R}^{k \cdot \text{poly}(\log(k/\delta))/\epsilon^2 \times k}$  that gives a subspace embedding of  $A$  with probability  $1 - \delta$ , i.e., with probability  $1 - \delta$ , we have :*

$$\|SAx\|_2 = (1 \pm \epsilon)\|Ax\|_2$$

holds for any  $x \in \mathbb{R}^n$ ,  $\|x\|_2 = 1$ . Moreover,  $SA$  can be computed in  $O(Nk \cdot \text{poly} \log k)$  time.

## 4 Our Algorithm

Our main algorithm is shown in Algorithm 1. We have the following convergence result of our algorithm.

► **Theorem 6.** *Suppose the width of a ReLU neural network satisfies*

$$m = \Omega(\max\{\lambda^{-4}n^4, \lambda^{-2}n^2d \log(16n/\delta)\}),$$

*then with probability  $1 - \delta$  over the random initialization of neural network and the randomness of the training algorithm, our algorithm (procedure FASTERTWOLAYER in Algorithm 1) achieves*

$$\|f_{t+1} - y\|_2 \leq \frac{1}{2}\|f_t - y\|_2.$$

*The computation cost in each iteration is  $\tilde{O}(mnd + n^3)$ , and the running time for reducing the training loss to  $\epsilon$  is  $\tilde{O}((mnd + n^3) \log(1/\epsilon))$ . Using fast matrix-multiplication, the total running time can be further reduced to  $\tilde{O}((mnd + n^\omega) \log(1/\epsilon))$ .*

■ **Algorithm 1** Faster algorithm for two-layer neural network.

---

1: <b>procedure</b> FASTERTWOLAYER()	▷ Theorem 6
2: $W_0$ is a random Gaussian matrix	▷ $W_0 \in \mathbb{R}^{md}$
3: <b>while</b> $t < T$ <b>do</b>	
4:         Compute the Jacobian matrix $J_t$	▷ $J_t \in \mathbb{R}^{n \times md}$
5:         Find an $\epsilon_0$ approximate solution using Algorithm 2	▷ $\epsilon_0 \in (0, \frac{1}{6}\sqrt{\lambda/n}]$
$\min_{g_t} \ J_t J_t^\top g_t - (f_t - y)\ _2 \tag{6}$	
6:         Update $W_{t+1} \leftarrow W_t - J_t^\top g_t$	
7: $t \leftarrow t + 1$	
8: <b>end while</b>	
9: <b>end procedure</b>	

---

The main difference between [23, 79] and our algorithm is that we perform an *approximate Newton update* (see line 6). The crucial observation here is that the Newton method is robust to small loss, thus it suffices to present a fine approximation. This observation is well-known in the convex optimization but unclear to the non-convex (but overparameterized) neural network setting. Another crucial observation is that instead of directly approximating the Gram matrix, it suffices to approximate  $(J_t J_t^\top)^{-1} g_t = G_t^{-1} g_t$ . Intuitively, this follows from

$$J_t^\top g_t \approx J_t (J_t J_t^\top)^{-1} (f_t - y) = (J_t^\top J_t)^\dagger J_t (f_t - y),$$

where  $(J_t^\top J_t)^\dagger$  denotes the pseudo-inverse of  $J_t^\top J_t$  and the last term is exactly the Newton update. This observation allows us to formulate the problem a regression problem (see Eq. (6)), on which we can introduce techniques from *randomize linear algebra* and develop fast algorithm that solves it in near linear time.

#### 4.1 Fast regression solver

■ **Algorithm 2** Fast regression.

---

```

1: procedure FASTREGRESSION( $A, \epsilon$ ) ▷ Lemma 7
2:   ▷  $A \in \mathbb{R}^{N \times k}$  is a full rank matrix,  $\epsilon \in (0, 1/2)$  is the desired precision
3:   Compute a subspace embedding  $SA$  ▷  $S \in \mathbb{R}^{k \text{poly}(\log k) \times k}$ 
4:   Compute  $R$  such that  $SAR$  orthonormal columns via QR decomposition ▷  $R \in \mathbb{R}^{k \times k}$ 
5:    $z_0 \leftarrow \vec{0} \in \mathbb{R}^k$ 
6:   while  $\|A^\top ARz_t - y\|_2 \geq \epsilon$  do
7:      $z_{t+1} \leftarrow z_t - (R^\top A^\top AR)^\top (R^\top A^\top ARz_t - R^\top y)$ 
8:   end while
9: return  $Rz_t$ 
10: end procedure

```

---

The core component of our algorithm is a fast regression solver (shown in Algorithm 2). The regression solver provides an approximate solution to  $\min_x \|A^\top Ax - y\|$  where  $A \in \mathbb{R}^{N \times k}$  ( $N \gg k$ ). We perform preconditioning on the matrix of  $A^\top A$  (line 3 – 4) and use gradient descent to derive an approximation solution (line 6 – 8).

► **Lemma 7.** *Let  $N = \Omega(k \text{poly}(\log k))$ . Given a matrix  $A \in \mathbb{R}^{N \times k}$ , let  $\kappa$  denote the condition number of  $A$ <sup>5</sup>, consider the following regression problem*

$$\min_{x \in \mathbb{R}^k} \|A^\top Ax - y\|_2. \quad (7)$$

Using procedure FASTREGRESSION (in Algorithm 2), with probability  $1 - \delta$ , we can compute an  $\epsilon$ -approximate solution  $x'$  satisfying

$$\|A^\top Ax' - y\|_2 \leq \epsilon \|y\|_2$$

in  $\tilde{O}(Nk \log(\kappa/\epsilon) + k^3)$  time.

**Speedup in Convex Optimization.** It should come as no surprise that our techniques can help accelerating a broad class of solvers in *convex optimization* problems as well. In the full version of this paper, we elaborate on this application, and in particular show how our technique improves the runtime of the “Newton-Sketch” algorithm of [59].

## 5 Conclusion and Open Problems

Our work provides a computationally-efficient (near-linear time) second-order algorithm for training sufficiently overparametrized two-layer neural network, overcoming the drawbacks of traditional first-order gradient algorithms. Our main technical contribution is developing a *faster regression solver* which uses linear sketching for fast preconditioning (in time

---

<sup>5</sup>  $\kappa = \sigma_{\max}(A)/\sigma_{\min}(A)$



independent of the network width). As such, our work demonstrates that the toolbox of randomized linear algebra can substantially reduce the computational cost of second-order methods in *non-convex optimization*, and not just in the convex setting for which it was originally developed (e.g., [59, 74, 25, 42, 40, 41, 47]).

Finally, we remark that, while the running time of our algorithm is  $\tilde{O}(Mn + n^3)$  (or  $O(Mn + n^\omega)$  using FMM), it is no longer (near) linear for networks with parameters  $M \leq n^2$  (resp.  $M \lesssim n^{\omega-1}$ ). While it is widely believed that  $\omega = 2$  [26], FMM algorithms are impractical at present, and it would therefore be very interesting to improve the extra additive term from  $n^3$  to  $n^{2+o(1)}$  (which seems best possible for dense  $n \times n$  matrices), or even to  $n^{3-\epsilon}$  using a practically viable algorithm. Faster preconditioners seem key to this avenue.

---

## References

---

- 1 Naman Agarwal, Brian Bullins, and Elad Hazan. Second-order stochastic optimization for machine learning in linear time. *The Journal of Machine Learning Research*, 18(1):4148–4187, 2017.
- 2 Nir Ailon and Bernard Chazelle. Approximate nearest neighbors and the fast johnson-lindenstrauss transform. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing (STOC)*, pages 557–563, 2006.
- 3 Zeyuan Allen-Zhu, Yuanzhi Li, and Yingyu Liang. Learning and generalization in overparameterized neural networks, going beyond two layers. In *Advances in neural information processing systems (NeurIPS)*, pages 6155–6166, 2019.
- 4 Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. A convergence theory for deep learning via over-parameterization. In *ICML*, volume 97, pages 242–252. PMLR, 2019. *arXiv version*: <https://arxiv.org/pdf/1811.03962.pdf>.
- 5 Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. On the convergence rate of training recurrent neural networks. In *NeurIPS*, pages 6673–6685, 2019. *arXiv version*: <https://arxiv.org/pdf/1810.12065.pdf>.
- 6 Alexandr Andoni, Chengyu Lin, Ying Sheng, Peilin Zhong, and Ruiqi Zhong. Subspace embedding and linear regression with orlicz norm. In *ICML*, pages 224–233. PMLR, 2018. *arXiv version*: <https://arxiv.org/pdf/1806.06430.pdf>.
- 7 Rohan Anil, Vineet Gupta, Tomer Koren, Kevin Regan, and Yoram Singer. Second order optimization made practical. *arXiv preprint*, 2020. *arXiv*:[arXiv:2002.09018](https://arxiv.org/abs/2002.09018).
- 8 Sanjeev Arora, Simon Du, Wei Hu, Zhiyuan Li, and Ruosong Wang. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *International Conference on Machine Learning (ICML)*, volume 97 of *Proceedings of Machine Learning Research*, pages 322–332. PMLR, 2019. *arXiv version*: <https://arxiv.org/pdf/1901.08584.pdf>.
- 9 Sanjeev Arora, Simon S Du, Wei Hu, Zhiyuan Li, Russ R Salakhutdinov, and Ruosong Wang. On exact computation with an infinitely wide neural net. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 8139–8148, 2019. *arXiv version*: <https://arxiv.org/pdf/1904.11955.pdf>.
- 10 Haim Avron, Michael Kapralov, Cameron Musco, Christopher Musco, Ameya Velingker, and Amir Zandieh. Random fourier features for kernel ridge regression: Approximation bounds and statistical guarantees. In *ICML*, volume 70, pages 253–262. PMLR, 2017. *arXiv version*: <https://arxiv.org/pdf/1804.09893.pdf>.
- 11 Haim Avron, Michael Kapralov, Cameron Musco, Christopher Musco, Ameya Velingker, and Amir Zandieh. A universal sampling method for reconstructing signals with simple fourier transforms. In *STOC*. ACM, 2019. *arXiv version*: <https://arxiv.org/pdf/1812.08723.pdf>. doi:10.1145/3313276.3316363.

- 12 Ainesh Bakshi, Nadiia Chepurko, and David P Woodruff. Robust and sample optimal algorithms for psd low-rank approximation. *arXiv preprint*, 2019. [arXiv:1912.04177](https://arxiv.org/abs/1912.04177).
- 13 Ainesh Bakshi, Rajesh Jayaram, and David P Woodruff. Learning two layer rectified neural networks in polynomial time. In *COLT*, volume 99, pages 195–268. PMLR, 2019. *arXiv version*: <https://arxiv.org/pdf/1811.01885.pdf>.
- 14 Ainesh Bakshi and David Woodruff. Sublinear time low-rank approximation of distance matrices. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 3782–3792, 2018. *arXiv version*: <https://arxiv.org/pdf/1809.06986.pdf>.
- 15 Frank Ban, David P. Woodruff, and Richard Zhang. Regularized weighted low rank approximation. In *NeurIPS*, pages 4061–4071, 2020. *arXiv version*: <https://arxiv.org/pdf/1911.06958.pdf>.
- 16 Sue Becker and Yann Le Cun. Improving the convergence of back-propagation learning with second order methods. In *Proceedings of the 1988 connectionist models summer school*, pages 29–37, 1988.
- 17 Alberto Bernacchia, Máté Lengyel, and Guillaume Hennequin. Exact natural gradient in deep linear networks and its application to the nonlinear case. In *Advances in Neural Information Processing Systems (NIPS)*, pages 5941–5950, 2018.
- 18 Aleksandar Botev, Hippolyt Ritter, and David Barber. Practical gauss-newton optimisation for deep learning. In *International Conference on Machine Learning (ICML)*, pages 557–565, 2017.
- 19 Christos Boutsidis and David P Woodruff. Optimal cur matrix decompositions. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC)*, pages 353–362. ACM, 2014. *arXiv version*: <https://arxiv.org/pdf/1405.7910.pdf>.
- 20 Christos Boutsidis, David P Woodruff, and Peilin Zhong. Optimal principal component analysis in distributed and streaming models. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing (STOC)*, pages 236–249, 2016.
- 21 Sébastien Bubeck. Convex optimization: Algorithms and complexity. *Foundations and Trends® in Machine Learning*, 8(3-4):231–357, 2015.
- 22 Sébastien Bubeck, Ronen Eldan, Yin Tat Lee, and Dan Mikulincer. Network size and weights size for memorization with two-layers neural networks. *arXiv preprint*, 2020. [arXiv:2006.02855](https://arxiv.org/abs/2006.02855).
- 23 Tianle Cai, Ruiqi Gao, Jikai Hou, Siyu Chen, Dong Wang, Di He, Zhihua Zhang, and Liwei Wang. A gram-gauss-newton method learning overparameterized deep neural networks for regression problems. *arXiv preprint*, 2019. [arXiv:1905.11675](https://arxiv.org/abs/1905.11675).
- 24 Kenneth L. Clarkson and David P. Woodruff. Low rank approximation and regression in input sparsity time. In *Symposium on Theory of Computing Conference (STOC)*, pages 81–90. ACM, 2013. *arXiv version*: <https://arxiv.org/pdf/1207.6365.pdf>.
- 25 Michael B Cohen, Yin Tat Lee, and Zhao Song. Solving linear programs in the current matrix multiplication time. In *STOC*, pages 938–942. ACM, 2019. *arXiv version*: <https://arxiv.org/pdf/1810.07896.pdf>.
- 26 Henry Cohn, Robert Kleinberg, Balazs Szegedy, and Christopher Umans. Group-theoretic algorithms for matrix multiplication. In *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 379–388. IEEE, 2005.
- 27 Amit Daniely. Memorizing gaussians with no over-parameterization via gradient decent on neural networks. *arXiv preprint*, 2020. [arXiv:2003.12895](https://arxiv.org/abs/2003.12895).
- 28 Huaian Diao, Rajesh Jayaram, Zhao Song, Wen Sun, and David Woodruff. Optimal sketching for kronecker product regression and low rank approximation. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 4739–4750, 2019. *arXiv version*: <https://arxiv.org/pdf/1909.13384.pdf>.
- 29 Petros Drineas, Malik Magdon-Ismail, Michael W Mahoney, and David P Woodruff. Fast approximation of matrix coherence and statistical leverage. *Journal of Machine Learning Research*, 13(Dec):3475–3506, 2012.

- 30 Petros Drineas, Michael W Mahoney, and Shan Muthukrishnan. Sampling algorithms for l2 regression and applications. In *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, pages 1127–1136. Society for Industrial and Applied Mathematics, 2006.
- 31 Simon S Du, Xiyu Zhai, Barnabas Poczos, and Aarti Singh. Gradient descent provably optimizes over-parameterized neural networks. In *ICLR*. OpenReview.net, 2019. *arXiv version*: <https://arxiv.org/pdf/1810.02054.pdf>.
- 32 John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research (JMLR)*, 12(Jul):2121–2159, 2011.
- 33 François Le Gall and Florent Urrutia. Improved rectangular matrix multiplication using powers of the coppersmith-winograd tensor. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1029–1046. SIAM, 2018. *arXiv version*: <https://arxiv.org/pdf/1708.05622.pdf>.
- 34 Thomas George, César Laurent, Xavier Bouthillier, Nicolas Ballas, and Pascal Vincent. Fast approximate natural gradient descent in a kronecker factored eigenbasis. In *Advances in Neural Information Processing Systems (NIPS)*, pages 9550–9560, 2018.
- 35 Roger Grosse and James Martens. A kronecker-factored approximate fisher matrix for convolution layers. In *International Conference on Machine Learning (ICML)*, pages 573–582, 2016.
- 36 Vineet Gupta, Tomer Koren, and Yoram Singer. Shampoo: Preconditioned stochastic tensor optimization. In *International Conference on Machine Learning (ICML)*, pages 1842–1850, 2018.
- 37 Moritz Hardt, Benjamin Recht, and Yoram Singer. Train faster, generalize better: Stability of stochastic gradient descent. In *ICML*, volume 48, pages 1225–1234. JMLR.org, 2016. *arXiv version*: <https://arxiv.org/pdf/1509.01240.pdf>.
- 38 Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems (NIPS)*, pages 8571–8580, 2018.
- 39 Ziwei Ji and Matus Telgarsky. Polylogarithmic width suffices for gradient descent to achieve arbitrarily small test error with shallow relu networks. In *ICLR*. OpenReview.net, 2020. *arXiv version*: <https://arxiv.org/pdf/1909.12292.pdf>.
- 40 Haotian Jiang, Tarun Kathuria, Yin Tat Lee, Swati Padmanabhan, and Zhao Song. A faster interior point method for semidefinite programming. In *Manuscript*, 2020.
- 41 Haotian Jiang, Yin Tat Lee, Zhao Song, and Sam Chiu-wai Wong. An improved cutting plane method for convex optimization, convex-concave games and its applications. In *STOC*, pages 944–953. ACM, 2020. *arXiv version*: <https://arxiv.org/pdf/2004.04250.pdf>.
- 42 Shunhua Jiang, Zhao Song, Omri Weinstein, and Hengjie Zhang. Faster dynamic matrix inverse for faster lps. *arXiv preprint*, 2020. *arXiv*:2004.07470.
- 43 Jonathan A Kelner, Lorenzo Orecchia, Aaron Sidford, and Zeyuan Allen Zhu. A simple, combinatorial algorithm for solving sdd systems in nearly-linear time. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing (STOC)*, pages 911–920. ACM, 2013. *arXiv version*: <https://arxiv.org/pdf/1301.6628.pdf>.
- 44 Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. *arXiv version*: <https://arxiv.org/pdf/1412.6980.pdf>.
- 45 Kasper Green Larsen and Jelani Nelson. Optimality of the johnson-lindenstrauss lemma. In *IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 633–638. IEEE, 2017. *arXiv version*: <https://arxiv.org/pdf/1609.02094.pdf>.
- 46 François Le Gall. Powers of tensors and fast matrix multiplication. In *Proceedings of the 39th international symposium on symbolic and algebraic computation (ISSAC)*, pages 296–303. ACM, 2014.
- 47 Yin Tat Lee, Zhao Song, and Qiuyi Zhang. Solving empirical risk minimization in the current matrix multiplication time. In *COLT*. <https://arxiv.org/pdf/1905.04447>, 2019.

- 48 Yuanzhi Li and Yingyu Liang. Learning overparameterized neural networks via stochastic gradient descent on structured data. In *NeurIPS*, pages 8168–8177, 2018. *arXiv version*: <https://arxiv.org/pdf/1808.01204.pdf>.
- 49 Yuanzhi Li and Yang Yuan. Convergence analysis of two-layer neural networks with ReLU activation. In *Advances in neural information processing systems (NIPS)*, pages 597–607, 2017. *arXiv version*: <https://arxiv.org/pdf/1705.09886.pdf>.
- 50 Hang Liao, Barak A. Pearlmutter, Vamsi K. Potluru, and David P. Woodruff. Automatic differentiation of sketched regression. In *AISTATS*, 2020.
- 51 Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. On the variance of the adaptive learning rate and beyond. *arXiv preprint*, 2019. [arXiv:1908.03265](https://arxiv.org/abs/1908.03265).
- 52 Yichao Lu, Paramveer Dhillon, Dean P Foster, and Lyle Ungar. Faster ridge regression via the subsampled randomized hadamard transform. In *Advances in neural information processing systems*, pages 369–377, 2013.
- 53 James Martens. Deep learning via hessian-free optimization. In *ICML*, volume 27, pages 735–742, 2010.
- 54 James Martens, Jimmy Ba, and Matthew Johnson. Kronecker-factored curvature approximations for recurrent neural networks. In *ICLR*, 2018.
- 55 James Martens and Roger Grosse. Optimizing neural networks with kronecker-factored approximate curvature. In *International conference on machine learning (ICML)*, pages 2408–2417, 2015.
- 56 Philipp Moritz, Robert Nishihara, and Michael Jordan. A linearly-convergent stochastic l-bfgs algorithm. In *Artificial Intelligence and Statistics*, pages 249–258, 2016.
- 57 Jelani Nelson and Huy L Nguyễn. Osnap: Faster numerical linear algebra algorithms via sparser subspace embeddings. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 117–126. IEEE, 2013. *arXiv version*: <https://arxiv.org/pdf/1211.1002.pdf>.
- 58 Samet Oymak and Mahdi Soltanolkotabi. Towards moderate overparameterization: global convergence guarantees for training shallow neural networks. *IEEE Journal on Selected Areas in Information Theory*, 2019.
- 59 Mert Pilanci and Martin J Wainwright. Newton sketch: A near linear-time optimization algorithm with linear-quadratic convergence. *SIAM Journal on Optimization*, 27(1):205–245, 2017.
- 60 Eric Price, Zhao Song, and David P. Woodruff. Fast regression with an  $\ell_\infty$  guarantee. In *International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 80 of *LIPICs*, pages 59:1–59:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. *arXiv version*: <https://arxiv.org/pdf/1705.10723.pdf>. doi:10.4230/LIPICs.ICALP.2017.59.
- 61 Ilya Razenshteyn, Zhao Song, and David P Woodruff. Weighted low rank approximations with provable guarantees. In *Proceedings of the 48th Annual Symposium on the Theory of Computing (STOC)*, 2016.
- 62 Vladimir Rokhlin and Mark Tygert. A fast randomized algorithm for overdetermined linear least-squares regression. *Proceedings of the National Academy of Sciences*, 105(36):13212–13217, 2008.
- 63 Tamás Sarlós. Improved approximation algorithms for large matrices via random projections. In *Proceedings of 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2006.
- 64 Zhao Song. *Matrix Theory : Optimization, Concentration and Algorithms*. PhD thesis, The University of Texas at Austin, 2019.
- 65 Zhao Song, Ruosong Wang, Lin Yang, Hongyang Zhang, and Peilin Zhong. Efficient symmetric norm regression via linear sketching. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 828–838, 2019. *arXiv version*: <https://arxiv.org/pdf/1910.01788.pdf>.

- 66 Zhao Song, David P Woodruff, and Peilin Zhong. Relative error tensor low rank approximation. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2772–2789. SIAM, 2019. *arXiv version*: <https://arxiv.org/pdf/1704.08246.pdf>.
- 67 Zhao Song and Xin Yang. Quadratic suffices for over-parametrization via matrix chernoff bound. *arXiv preprint*, 2019. **arXiv:1906.03593**.
- 68 Daniel A. Spielman and Shang-Hua Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *Proceedings of the Thirty-sixth Annual ACM Symposium on Theory of Computing (STOC)*, pages 81–90. ACM, 2004.
- 69 Joel A Tropp. Improved analysis of the subsampled randomized hadamard transform. *Advances in Adaptive Data Analysis*, 3(01n02):115–126, 2011.
- 70 Pravin M Vaidya. A new algorithm for minimizing convex functions over convex sets. In *30th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 338–343. IEEE, 1989.
- 71 Pravin M Vaidya. Speeding-up linear programming using fast matrix multiplication. In *30th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 332–337. IEEE, 1989.
- 72 Ruosong Wang and David P Woodruff. Tight bounds for  $\ell_p$  oblivious subspace embeddings. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1825–1843. SIAM, 2019. *arXiv version*: <https://arxiv.org/pdf/1801.04414.pdf>.
- 73 Virginia Vassilevska Williams. Multiplying matrices faster than coppersmith-winograd. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing (STOC)*, pages 887–898. ACM, 2012.
- 74 David P. Woodruff. Sketching as a tool for numerical linear algebra. *Foundations and Trends in Theoretical Computer Science*, 10(1-2):1–157, 2014.
- 75 David P. Woodruff and Amir Zandieh. Near input sparsity time kernel embeddings via adaptive sampling. In *ICML*, 2020.
- 76 David P Woodruff and Peilin Zhong. Distributed low rank approximation of implicit functions of a matrix. In *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*, pages 847–858. IEEE, 2016.
- 77 Xiaoxia Wu, Simon S Du, and Rachel Ward. Global convergence of adaptive gradient methods for an over-parameterized neural network. *arXiv preprint*, 2019. **arXiv:1902.07111**.
- 78 Yuhuai Wu, Elman Mansimov, Roger B Grosse, Shun Liao, and Jimmy Ba. Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation. In *Advances in neural information processing systems (NIPS)*, pages 5279–5288, 2017.
- 79 Guodong Zhang, James Martens, and Roger B Grosse. Fast convergence of natural gradient descent for over-parameterized neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 8080–8091, 2019.
- 80 Kai Zhong, Zhao Song, and Inderjit S Dhillon. Learning non-overlapping convolutional neural networks with multiple kernels. *arXiv preprint*, 2017. **arXiv:1711.03440**.
- 81 Kai Zhong, Zhao Song, Prateek Jain, Peter L. Bartlett, and Inderjit S. Dhillon. Recovery guarantees for one-hidden-layer neural networks. In *ICML*, volume 70, pages 4140–4149. PMLR, 2017. *arXiv version*: <https://arxiv.org/pdf/1706.03175.pdf>.





# A New Connection Between Node and Edge Depth Robust Graphs

Jeremiah Blocki

Department of Computer Science, Purdue University, West Lafayette, IN, USA  
jblocki@purdue.edu

Mike Cinkoske

Department of Computer Science, University of Illinois Urbana-Champaign, IL, USA  
mjc18@illinois.edu

---

## Abstract

Given a directed acyclic graph (DAG)  $G = (V, E)$ , we say that  $G$  is  $(e, d)$ -depth-robust (resp.  $(e, d)$ -edge-depth-robust) if for any set  $S \subset V$  (resp.  $S \subseteq E$ ) of at most  $|S| \leq e$  nodes (resp. edges) the graph  $G - S$  contains a directed path of length  $d$ . While edge-depth-robust graphs are potentially easier to construct many applications in cryptography require node depth-robust graphs with small indegree. We create a graph reduction that transforms an  $(e, d)$ -edge-depth-robust graph with  $m$  edges into a  $(e/2, d)$ -depth-robust graph with  $O(m)$  nodes and constant indegree. One immediate consequence of this result is the first construction of a provably  $(\frac{n \log \log n}{\log n}, \frac{n}{(\log n)^{1+\log \log n}})$ -depth-robust graph with constant indegree, where previous constructions for  $e = \frac{n \log \log n}{\log n}$  had  $d = O(n^{1-\epsilon})$ . Our reduction crucially relies on ST-Robust graphs, a new graph property we introduce which may be of independent interest. We say that a directed, acyclic graph with  $n$  inputs and  $n$  outputs is  $(k_1, k_2)$ -ST-Robust if we can remove any  $k_1$  nodes and there exists a subgraph containing at least  $k_2$  inputs and  $k_2$  outputs such that each of the  $k_2$  inputs is connected to all of the  $k_2$  outputs. If the graph is  $(k_1, n - k_1)$ -ST-Robust for all  $k_1 \leq n$  we say that the graph is maximally ST-robust. We show how to construct maximally ST-robust graphs with constant indegree and  $O(n)$  nodes. Given a family  $\mathbb{M}$  of ST-robust graphs and an arbitrary  $(e, d)$ -edge-depth-robust graph  $G$  we construct a new constant-indegree graph  $\text{Reduce}(G, \mathbb{M})$  by replacing each node in  $G$  with an ST-robust graph from  $\mathbb{M}$ . We also show that ST-robust graphs can be used to construct (tight) proofs-of-space and (asymptotically) improved wide-block labeling functions.

**2012 ACM Subject Classification** Security and privacy  $\rightarrow$  Mathematical foundations of cryptography; Theory of computation  $\rightarrow$  Cryptographic primitives

**Keywords and phrases** Depth robust graphs, memory hard functions

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.64

**Related Version** A full version of the paper is available at <https://arxiv.org/pdf/1910.08920.pdf>.

**Funding** This research was supported by the National Science Foundation under awards CNS #1755708 and CNS #1704587.

*Mike Cinkoske:* Supported by NSF REU #1934444.

## 1 Introduction

Given a directed acyclic graph (DAG)  $G = (V, E)$ , we say that  $G$  is  $(e, d)$ -reducible (resp.  $(e, d)$ -edge reducible) if there is a subset  $S \subseteq V$  (resp.  $S \subseteq E$ ) of  $|S| \leq e$  nodes (resp. edges) such that  $G - S$  does not contain a directed path of length  $d$ . If a graph is not  $(e, d)$ -reducible (resp.  $(e, d)$ -edge reducible) we say that the graph is  $(e, d)$ -depth robust (resp.  $(e, d)$ -edge-depth-robust). Depth robust graphs have found many applications in the field of cryptography in the construction of proofs of sequential work [11], proofs of space [7, 12], and in the construction of data independent memory hard functions (iMHFs). For example, highly depth robust graphs are known to be necessary [1] and sufficient [3] to construct



© Jeremiah Blocki and Mike Cinkoske;

licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 64; pp. 64:1–64:18

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



iMHF's with high amortized space time complexity. While edge depth-robust graphs are often easier to construct [14], most applications require node depth-robust graphs with small indegree.

It has been shown [17] that in any DAG with  $m$  edges and  $n$  nodes and any  $i \leq \log_2 n$ , there exists a set  $S_i$  of  $\frac{mi}{\log n}$  edges that will destroy all paths of length  $n/2^i$  forcing  $\text{depth}(G - S_i) \leq \frac{n}{2^i}$ . For DAGs with constant indegree we have  $m = O(n)$  edges so an equivalent condition holds for node depth robustness [1], since a node can be removed by removing all the edges incident to it. In particular, there exists a set  $S_i$  of  $O(\frac{ni}{\log n})$  nodes such that  $\text{depth}(G - S_i) \leq \frac{n}{2^i}$  for all  $i < \log n$ . It is known how to construct an  $(c_1 n / \log n, c_2 n)$ -depth-robust graph, for suitable  $c_1, c_2 > 0$  [3] and an  $(c_3 n, c_4 n^{1-\epsilon})$ -depth-robust graph for small  $\epsilon$  for [14].

An open challenge is to construct constant indegree  $(c_1 ni / \log n, c_2 n / 2^i)$ -depth-robust graphs which match the Valiant bound [17] for intermediate values of  $i = \omega(1)$  and  $i = o(\log n)$ . For example, when  $i = \log \log n$  then the Valiant bound [17] does not rule out the existence of  $(c_1 ni / \log n, c_2 n / \log n)$ -depth-robust graphs with constant indegree. Such a graph would yield asymptotically stronger iMHFs [5]. While there are several constructions that are conjectured to be  $(c_1 ni / \log n, c_2 n / \log n)$ -depth-robust the best provable lower bound for  $(e = cni / \log n, d)$ -depth robustness of a constant indegree graph is  $d = \Omega(n^{1-\epsilon})$ . For edge-depth robustness we have constructions of graphs with  $m = O(n \log n)$  edges which are  $(e_i, d_i)$ -edge depth robust for any  $i$  with  $e_i = mi / \log n$  and  $d_i = n / \log^{i+1} n$  — much closer to matching the Valiant bound [17].

## 1.1 Contributions

Our main contribution is a graph reduction that transforms any  $(e, d)$ -edge-depth-robust graph with  $m$  edges into a  $(e/2, d)$ -depth-robust graph with  $O(m)$  nodes and constant indegree. Our reduction utilizes ST-Robust graphs, a new graph property we introduce and construct. We believe that ST-Robust graphs may be of independent interest.

Intuitively, a  $(k_1, k_2)$ -ST-Robust graph with  $n$  inputs  $I$  and  $n$  outputs  $O$  satisfies the property that, even after deleting  $k_1$  nodes from the graph we can find  $k_2$  inputs  $x_1, \dots, x_{k_2}$  and  $k_2$  outputs  $y_1, \dots, y_{k_2}$  such that *every* input  $x_i$  ( $i \in [k_2]$ ) is still connected to *every* output  $y_j$  ( $j \in [k_2]$ ). If we can guarantee that the each directed path from  $x_i$  to  $y_j$  has length  $d$  then we say that the graph is  $(k_1, k_2, d)$ -ST-Robust. A maximally depth-robust graph should be  $(k_1, n - k_1)$ -depth robust for any  $k_1$ .

► **Definition 1 (ST-Robust).** Let  $G = (V, E)$  be a DAG with  $n$  inputs, denoted by set  $I$  and  $n$  outputs, denoted by set  $O$ . Then  $G$  is  $(k_1, k_2)$ -ST-robust if  $\forall D \subset V(G)$  with  $|D| \leq k_1$ , there exists subgraph  $H$  of  $G - D$  with  $|I \cap V(H)| \geq k_2$  and  $|O \cap V(H)| \geq k_2$  such that  $\forall s \in I \cap V(H)$  and  $\forall t \in O \cap V(H)$  there exists a path from  $s$  to  $t$  in  $H$ . If  $\forall s \in I \cap V(H)$  and  $\forall t \in O \cap V(H)$  there exists a path from  $s$  to  $t$  of length  $\geq d$  then we say that  $G$  is  $(k_1, k_2, d)$ -ST-robust.

► **Definition 2 (Maximally ST-Robust).** Let  $G = (V, E)$  be a constant indegree DAG with  $n$  inputs and  $n$  outputs. Then  $G$  is  $c_1$ -maximally ST-robust (resp.  $c_1$  max ST-robust with depth  $d$ ) if there exists a constant  $0 < c_1 \leq 1$  such that  $G$  is  $(k, n - k)$ -ST-robust (resp.  $(k, n - k, d)$ -ST-robust) for all  $k$  with  $0 \leq k \leq c_1 n$ . If  $c_1 = 1$ , we just say that  $G$  is maximally ST-Robust.

We show how to construct maximally ST-robust graphs with constant indegree and  $O(n)$  nodes and we show how maximally ST-robust graphs can be used to transform any  $(e, d)$ -edge-depth-robust graph  $G$  with  $m$  edges into a  $(e/2, d)$ -depth-robust graph  $G'$  with

$O(m)$  nodes and constant indegree. Intuitively, in our reduction each node  $v \in V(G)$  with degree  $\delta(v)$  is replaced with a maximally ST-robust graph  $M_{\delta(v)}$  with  $\delta(v)$  inputs/outputs. Incoming edges into  $v$  are redirected into the inputs  $I_{\delta(v)}$  of the ST-robust graph. Similarly,  $v$ 's outgoing edges are redirected out of the outputs  $O_{\delta(v)}$  of the ST-robust graph. Because  $M_{\delta(v)}$  is maximally ST-robust, when a node is removed from  $M_{\delta(v)}$  the set of inputs and outputs where each input connects to every output has at most one input and one output node removed. Each input or output node removed from  $M_{\delta(v)}$  corresponds to removing at most one edge from the original graph. Thus, removing  $k$  nodes from  $M_{\delta(v)}$  corresponds to destroying at most  $2k$  edges in the original graph  $G$ .

Our reduction gives us a fundamentally new way to design node-depth-robust graphs: design an edge-depth-robust graph (easier) and then reduce it to a node-depth-robust graph. The reduction can be used with a construction from [14] to construct a  $(\frac{n \log \log n}{\log n}, \frac{n}{(\log n)^{1+\log \log n}})$ -depth-robust graph. We conjecture that several prior DAG constructions (e.g., [4, 8, 14]) are actually  $(n \log \log n, \frac{n}{\log n})$ -edge-depth-robust. If any of these conjectures are true then our reduction would immediately yield the desired  $(\frac{n \log \log n}{\log n}, \frac{n}{\log n})$ -depth-robust graph.

We also present several other applications for maximally ST-robust graphs including (tight) proofs-of-space and wide block-labeling functions.

## 2 Edge to Node Depth-Robustness

In this section, we use the fact that linear sized, constant indegree, maximally ST-robust graphs exist to construct a transformation of an  $(e, d)$ -edge-depth robust graph with  $m$  edges into an  $(e, d)$ -node-depth robust graph with constant indegree and  $O(m)$  nodes. In the next section we will construct a family of ST-robust graphs that satisfies Theorem 3.

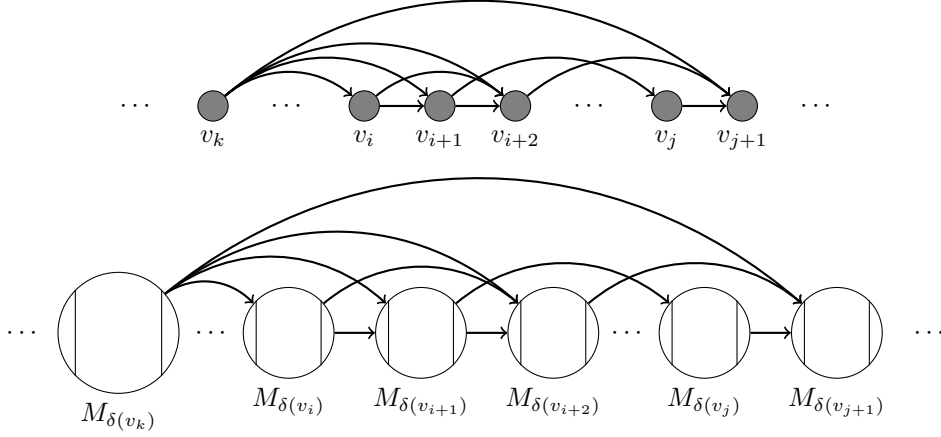
► **Theorem 3** (Key Building Block). *There exists a family of graphs  $\mathbb{M} = \{M_n\}_{n=1}^{\infty}$  with the property that for each  $n \geq 1$ ,  $M_n$  has constant indegree,  $O(n)$  nodes, and is maximally ST-Robust.*

### 2.1 Reduction Definition

Let  $G = (V, E)$  be a DAG, and let  $\mathbb{M}$  be as in Theorem 3. Then we define  $\text{Reduce}(G, \mathbb{M})$  in construction 4 as follows:

► **Construction 4** ( $\text{Reduce}(G, \mathbb{M})$ ). *Let  $G = (V, E)$  and let  $\mathbb{M}$  be the family of graphs defined above. For each  $M_n \in \mathbb{M}$ , we say that  $M_n = (V(M_n), E(M_n))$ , with  $V(M_n) = I(M_n) \cup O(M_n) \cup D(M_n)$ , where  $I(M_n)$  are the inputs of  $M_n$ ,  $O(M_n)$  are the outputs, and  $D(M_n)$  are the internal vertices. For  $v \in V$ , let  $\delta(v) = \max\{\text{indegree}(v), \text{outdegree}(v)\}$ . Then we define  $\text{Reduce}(G) = (V_R, E_R)$ , where  $V_R = \{(v, w) \mid v \in V, w \in V(M_{\delta(v)})\}$  and  $E_R = E_{\text{internal}} \cup E_{\text{external}}$ . We let  $E_{\text{internal}} = \{((v, u'), (v, w')) \mid v \in V, (u', w') \in E(M_{\delta(v)})\}$ . Then for each  $v \in V$ , we define an  $\text{In}(v) = \{u : (u, v) \in E\}$  and  $\text{Out}(v) = \{u : (v, u) \in E\}$  and then pick two injective mappings  $\pi_{\text{in}, v} : \text{In}(v) \rightarrow I(V(M_{\delta(v)}))$  and  $\pi_{\text{out}, v} : \text{Out}(v) \rightarrow O(V(M_{\delta(v)}))$ . We let  $E_{\text{external}} = \{((u, \pi_{\text{out}, u}(v)), (v, \pi_{\text{in}, v}(u))) : (u, v) \in E\}$ .*

*Intuitively, to construct  $\text{Reduce}(G, \mathbb{M})$  we replace every node of  $G$  with a constant indegree, maximally ST-robust graph, mapping the edges connecting two nodes from the outputs of one ST-robust graph to the inputs of another. Then for every  $e = (u, w) \in E$ , add an edge from an output of  $M_{\delta(u)}$  to an input of  $M_{\delta(w)}$  such that the outputs of  $M_{\delta(u)}$  have outdegree at most 1, and the inputs of  $M_{\delta(w)}$  have indegree at most 1. If  $v \in V$  is replaced by  $M_{\delta(v)}$ , then we call  $v$  the genesis node and  $M_{\delta(v)}$  its metanode.*



■ **Figure 1** Diagram of the transformation  $\text{Reduce}(G, \mathbb{M})$ .

## 2.2 Proof of Main Theorem

We now state the main result of this section which says that if  $G$  is edge-depth robust then  $\text{Reduce}(G, \mathbb{M})$  is node depth-robust.

► **Theorem 5.** *Let  $G$  be an  $(e, d)$ -edge-depth-robust DAG with  $m$  edges. Let  $\mathbb{M}$  be a family of max ST-Robust graphs with constant indegree. Then  $G' = (V', E') = \text{Reduce}(G, \mathbb{M})$  is  $(e/2, d)$ -depth robust. Furthermore,  $G'$  has maximum indegree  $\max_{v \in V(G)} \{\text{indeg}(M_{\delta(v)})\}$ , and its number of nodes is  $\sum_{v \in V(G)} |V(M_{\delta(v)})|$  where  $\delta(v) = \max\{\text{indeg}(v), \text{outdeg}(v)\}$ .*

A formal proof can be found in Appendix B. We briefly outline the intuition for this proof below.

**Proof (Intuition).** The first thing we note is that each graph  $M_{\delta(v)}$  has constant indegree at most  $c\delta(v)$  nodes for some constant  $c > 0$ . Therefore, the graph  $G'$  has  $\sum_{v \in V(G)} |V(M_{\delta(v)})| \leq c \sum_v \delta(v) \leq 2cm$  nodes and  $G'$  has constant indegree.

Now for any set  $S \subseteq V'$  of nodes we remove from  $G'$  we will map  $S$  to a corresponding set  $S_{irr} \subseteq E$  of at most  $|S_{irr}| \leq 2|S|$  irreparable edges in  $G$ . We then prove that any path  $P$  in  $G - S_{irr}$  corresponds to a longer path  $P'$  in  $G' - S$  that is at least as long. Intuitively, each incoming edge  $(u, v)$  (resp. outgoing edge  $(v, w)$ ) in  $E(G)$  corresponds to an input node (resp. output node) in  $v$ 's corresponding metanode  $M_{\delta(v)}$  which we will label  $x_{u,v}$  (resp.  $y_{v,w}$ ). If  $S \subseteq V'$  removes at most  $k$  nodes from the metanode  $M_{\delta(v)}$  then, by maximal ST-robustness, we still can find  $\delta(v) - k$  inputs and  $\delta(v) - k$  outputs that are *all* pairwise connected. If  $x_{u,v}$  (resp.  $y_{v,w}$ ) is not part of this pairwise connected subgraph then we will add the corresponding edge  $(u, v)$  (resp.  $(v, w)$ ) to the set  $S_{irr}$ . Thus, the set  $S_{irr}$  will have size at most  $2|S|$  (Claim 30 in the appendix).

Intuitively, any path  $P$  in  $G - S_{irr}$  can be mapped to a longer path  $P'$  in  $G' - S$  (Claim 29). If  $P$  contains the edges  $(u, v), (v, w)$  then we know that the input node  $x_{u,v}$  and output node  $y_{v,w}$  node in  $M_{\delta(v)}$  are still connected in  $G' - S$ . ◀

► **Corollary 6** (of Theorem 5). *If there exists some constants  $c_1, c_2$ , such that we have a family  $\mathbb{M} = \{M_n\}_{n=1}^{\infty}$  of linear sized  $|V(M_n)| \leq c_1 n$ , constant indegree  $\text{indeg}(M_n) \leq c_2$ , and maximally ST-Robust graphs, then  $\text{Reduce}(G, \mathbb{M})$  has maximum indegree  $c_2$  and the number of nodes is at most  $2c_1 m$ .*

The next corollary states that if we have a family of maximally ST-Robust graphs with  $\mathbb{M} = \{M_k\}_{k=1}^\infty$  depth  $d_k$  then we can transform any  $(e, d)$ -edge-depth-robust DAG  $G = (V, E)$  with maximum degree  $\delta = \max_{v \in V} \delta(v)$  into  $(e/2, d \cdot d_\delta)$ -depth robust graph. Instead of replacing each node  $v \in G$  with a copy of  $M_{\delta(v)}$ , we instead replace each node with a copy of  $M_{\delta, v} := M_\delta$ , attaching the edges same way as in Construction 4. Thus the transformed graph  $G'$  has  $|V(G)| \times |M_\delta|$  nodes and constant indegree. Intuitively, any path  $P$  of length  $d$  in  $G - S_{irr}$  now maps to a path  $P'$  of length  $d \times d_\delta$  – if  $P$  contains the edges  $(u, v), (v, w)$  then we know that the input node  $x_{u,v}$  and output node  $y_{u,v}$  node in  $M_{\delta, v}$  are connected in  $G' - S$  by a path of length at least  $d_\delta$ .

► **Corollary 7** (of Theorem 5). *Suppose that there exists a family  $\mathbb{M} = \{M_k\}_{k=1}^\infty$  of max ST-Robust graphs with depth  $d_k$  and constant indegree. Given any  $(e, d)$ -edge-depth-robust DAG  $G$  with  $n$  nodes and maximum degree  $\delta$  we can construct a DAG  $G'$  with  $n \times |M_\delta|$  nodes and constant indegree that is  $(e/2, d \cdot d_\delta)$ -depth robust.*

**Proof (sketch).** Instead of replacing each node  $v \in G$  with a copy of  $M_{\delta(v)}$ , we instead replace each node with a copy of  $M_{\delta, v} := M_\delta$ , attaching the edges same way as in Construction 4. Thus the transformed graph  $G'$  has  $|V(G)| \times |M_\delta|$  nodes and constant indegree. Let  $S \subset V(G')$  be a set of nodes that we will remove from  $G'$ . By Claim 29, there exists a path  $P$  in  $G' - S$  that passes through  $d$  metanodes  $M_{\delta, v_1}, \dots, M_{\delta, v_d}$ . Since  $M_\delta$  is maximally ST-robust with depth  $d_\delta$  the sub-path  $P_i = P \cap M_{\delta, v_i}$  through each metanode has length  $|P_i| \geq d_\delta$ . Thus, the total length of the path is at least  $\sum_i |P_i| \geq d \cdot d_\delta$ . ◀

► **Corollary 8** (of Theorem 5). *Let  $\epsilon > 0$  be any fixed constant. Given any family  $\{G_m\}_{m=1}^\infty$  of  $(e_m, d_m)$ -edge-depth-robust DAGs  $G_m$  with  $m$  nodes and maximum indegree  $\delta_m$  then for some constants  $c_1, c_2 > 0$  we can construct a family  $\{H_m\}_{m=1}^\infty$  of DAGs such that each DAG  $H_m$  is  $(e_m/2, d_m \cdot \delta_m^{1-\epsilon})$ -depth robust,  $H_m$  has maximum indegree at most  $c_2$  (constant) and at most  $|V(H_m)| \leq c_1 m \delta_m$  nodes.*

**Proof (sketch).** This follows immediately from Corollary 7 and from our construction of a family  $\mathbb{M}_\epsilon = \{M_{k, \epsilon}\}_{k=1}^\infty$  of max ST-Robust graphs with depth  $d_k > k^{1-\epsilon}$  and constant indegree. ◀

► **Corollary 9** (of Theorem 5). *Let  $\{e_m\}_{m=1}^\infty$  and  $\{d_m\}_{m=1}^\infty$  be any sequence. If there exists a family  $\{G_m\}_{m=1}^\infty$  of  $(e_m, d_m)$ -edge-depth-robust graphs, where each DAG  $G_m$  has  $m$  edges, then there is a corresponding family  $\{H_n\}_{n=1}^\infty$  of constant indegree DAGs such that each  $H_n$  has  $n$  nodes and is  $(\Omega(e_n), \Omega(d_n))$ -depth-robust.*

The original Grate's construction [14],  $G$ , has  $N = 2^n$  nodes and  $m = n2^n$  edges and for any  $s \leq n$ , and is  $(s2^n, \sum_{j=0}^s \frac{N}{\binom{n}{j}})$ -edge-depth-robust. For node depth-robustness we only had matching constructions when  $s = O(1)$  [2, 3] and  $s = \Omega(\log N)$  [14] – no comparable lower bounds were known for intermediate  $s$ .

► **Corollary 10** (of Theorem 5). *There is a family of constant indegree graphs  $\{G_n\}$  such that  $G_n$  has  $O(N = 2^n)$  nodes and  $G_n$  is  $(sN/(2n), \sum_{j=0}^s \frac{N}{\binom{n}{j}})$ -edge-depth-robust for any  $1 \leq s \leq \log n$*

In particular, setting  $s = \log \log n$  and applying the indegree reduction from Theorem 5, we see that the transformed graph  $G'$  has constant indegree,  $N' = O(n2^n)$  nodes, and is  $(\frac{N' \log \log N'}{\log N'}, \frac{N'}{(\log N')^{1+\log \log N'}})$ -depth-robust. Blocki et al. [5] showed that if there exists a node depth robust graph with  $e = \Omega(N \log \log N / \log N)$  and  $d = \Omega(N \log \log N / \log N)$  then

one can obtain another constant indegree graph with pebbling cost  $\Omega(N^2 \log \log N / \log N)$  which is optimal for constant indegree graphs. We conjecture that the graphs in [8] are sufficiently edge depth robust to meet these bounds after being transformed by our reduction.

### 3 ST Robustness

In this section we show how to construct maximally ST-robust graphs with constant indegree and linear size. We first introduce some of the technical building blocks used in our construction including superconcentrators [10, 13, 16] and grates [14]. Using these building blocks we then provide a randomized construction of a  $c_1$ -maximally ST-robust DAG with linear size and constant indegree for some constant  $c_1 > 0$  – sampled graphs are  $c_1$ -maximally ST-robust DAG with high probability. Finally, we use  $c_1$ -maximally ST-robust DAGs to construct a family of maximally ST-robust graphs with linear size and constant indegree.

#### 3.1 Technical Ingredients

We now introduce other graph properties that will be useful for constructing ST-robust graphs.

##### Grates

A DAG  $G = (V, E)$  with  $n$  inputs  $I$  and  $n$  outputs  $O$  is called a  $(c_0, c_1)$ -grate if for any subset  $S \subset V$  of size  $|S| \leq c_0 n$  at least  $c_1 n^2$  input output pairs  $(x, y) \in I \times O$  remain connected by a directed path from  $x$  to  $y$  in  $G - S$ . Schnitger [14] showed how to construct  $(c_0, c_1)$ -grates with  $O(n)$  nodes and constant indegree for suitable constants  $c_0, c_1 > 0$ . The notion of an maximally ST-robust graph is a strictly stronger requirement since there is no requirement on which pairs are connected. However, we show that a slight modification of Schnitger’s [14] construction is a  $(cn, n/2)$ -ST-robust for a suitable constant  $c$ . We then transform this graph into a  $c_1$ -maximally ST-robust graph by sandwiching it in between two superconcentrators. Finally, we show how to use several  $c_1$ -maximally ST-robust graphs to construct a maximally ST-robust graph.

##### Superconcentrators

We say that a directed acyclic graph  $G = (V, E)$  with  $n$  input vertices and  $n$  output vertices is an  **$n$ -superconcentrator** if for any  $r$  inputs and any  $r$  outputs,  $1 \leq r \leq n$ , there are  $r$  vertex-disjoint paths in  $G$  connecting the set of these  $r$  inputs to these  $r$  outputs. We note that there exists linear size, constant indegree superconcentrators [10, 13, 16] and we use this fact throughout the rest of the paper. For example, Pippenger [13] constructed an  $n$ -superconcentrator with at most  $41n$  vertices and indegree at most 16.

##### Connectors

We say that an  $n$ -superconcentrator is an  **$n$ -connector** if it is possible to specify which input is to be connected to which output by vertex disjoint paths in the subsets of  $r$  inputs and  $r$  outputs. Connectors and superconcentrators are potential candidates for ST-robust graphs because of their highly connective properties. In fact, we can prove that any connectors  **$n$ -connector** is maximally ST-robust – the proof of Theorem 11 can be found in the appendix. While we have constructions of  **$n$ -connector** graphs these graphs have  $O(n \log n)$  vertices and indegree of 2, an information theoretic technique of Shannon [15] can be used to

prove that any  $n$ -connector with constant indegree requires *at least*  $\Omega(n \log n)$  vertices – see discussion in the appendix. Thus, we cannot use  $n$ -connectors to build linear sized ST-robust graphs. However, Shannon’s information theoretic argument does not rule out the existence of linear size ST-robust graphs.

► **Theorem 11.** *If  $G$  is an  $n$ -connector, then  $G$  is  $(k, n - k)$ -ST-robust, for all  $1 \leq k \leq n$ .*

### 3.2 Linear Size ST-robust Graphs

ST-robust graphs have similar connective properties to connectors, so a natural question to ask is whether ST-robust graphs with constant indegree require  $\Omega(n \log n)$  vertices. In this section, we show that linear size ST-robust graphs exist by showing that a modified version of the Grates construction [14] becomes  $c$ -maximally ST-robust when sandwiched between two superconcentrators for some constant  $c$ .

In the proof of Theorem A in [14], Schnitger constructs a family of DAGs  $(H_n | n \in N)$  with constant indegree  $\delta_H$ , where  $n$  is the number of nodes and  $H_n$  is  $(cn, n^{2/3})$ -depth-robust, for suitable constant  $c > 0$ . We construct a similar graph  $G_n$  as follows:

► **Construction 12** ( $G_n$ ). *We begin with  $H_n^1, H_n^2$  and  $H_n^3$ , three isomorphic copies of  $H_n$  with disjoint vertex sets  $V_1, V_2$  and  $V_3$ . For each top vertex  $v \in V_3$  sample  $\tau$  vertices  $x_1^v, \dots, x_\tau^v$  independently and uniformly at random from  $V_2$  and for each  $i \leq \tau$  add each directed edge  $(x_i^v, v)$  to  $G_n$  to connect each of these sampled nodes to  $v$ . Similarly, for each node vertex  $u \in V_2$  sample  $\tau$  vertices  $x_1^u, \dots, x_\tau^u$  from  $V_1$  independently and uniformly at random and add each directed edge  $(x_i^u, u)$  to  $G_n$ . Note that  $\text{indeg}(G_n) \leq \text{indeg}(H_n) + \tau$ .*

Schnitger’s construction only utilizes two isomorphic copies of  $H_n$  and the edges connecting  $H_n^1$  and  $H_n^2$  a sampled by picking  $\tau$  random permutations. In our case the analysis is greatly simplified by picking the edges uniformly and we will need three layers to prove ST-robustness. We will use the following lemma from the Grates paper as a building block. A proof of Lemma 13 is included in the appendix for completeness.

► **Lemma 13** ([14]). *For some suitable constant  $c > 0$  any any subset  $S$  of  $cn/2$  vertices of  $G_n$  the graph  $H_n^1 - S$  contains  $k = cn^{1/3}/2$  vertex disjoint paths  $A_1, \dots, A_k$  of length  $n^{2/3}$  and  $H_n^2 - S$  contains  $k$  vertex disjoint paths  $B_1, \dots, B_k$  of the same length.*

We use Lemma 13 to help establish our main technical Lemma 14. We sketch the proof of Lemma 14 below. A formal proof can be found in Appendix B.

► **Lemma 14.** *Let  $G_n$  be defined as in Construction 12. Then for some constants  $c > 0$ , with high probability  $G_n$  has the property that for all  $S \subset V(G_n)$  with  $|S| = cn/2$  there exists  $A \subseteq V(H_n^1)$  and  $B \subseteq V(H_n^3)$  such that for every pair of nodes  $u \in A$  and  $v \in B$  the graph  $G_n - S$  contains a path from  $u$  to  $v$  and  $|A|, |B| \geq 9cn/40$ .*

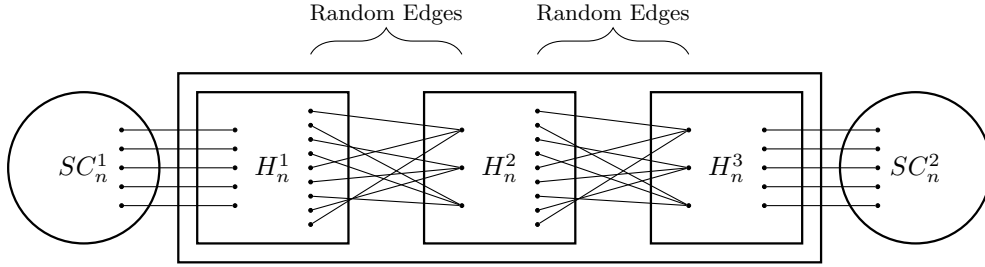
**Proof (sketch).** Fixing any  $S$  we can apply Lemma 13 to find  $k := cn^{1/3}/2$  vertex disjoint paths  $P_{1,S}^i, \dots, P_{k,S}^i$  in  $H_n^i$  of length  $n^{2/3}$  for each  $i \leq 3$ . Here,  $c$  is the constant from Lemma 13. Let  $U_{j,S}^i$  be the upper half of the  $j$ -th path in  $H_n^i$  and  $L_{j,S}^i$  be the lower half and define the event  $BAD_{i,S}^{upper}$  to be the event that there exists at least  $k/10$  indices  $j \leq k$  s.t.,  $U_{j,S}^i$  is disconnected from  $L_{i,S}^3$ . We construct  $B$  by taking the union of all of upper paths  $U_{i,S}^3$  in  $H_n^3$  for each non-bad (upper) indices  $i$ . Similarly, we define  $BAD_{i,S}^{lower}$  to be the event that there exists at least  $k/10$  indices  $j \leq k$  s.t.  $U_{i,S}^1$  is disconnected from  $L_{j,S}^2$  and we construct  $A$  by taking the union of all of the lower paths  $L_{i,S}^1$  in  $H_n^1$  for each non-bad (lower) indices  $i$ . We can now argue that any pair of nodes  $u \in A$  and  $v \in B$  is connected by invoking



the pigeonhole principle i.e., if  $u \in L_{i,S}^1$  and  $v \in U_{i',S}^3$  for good indices  $i$  and  $i'$  then there exists some path  $P_j^2$  in the middle layer  $H_n^2$  which can be used to connect  $u$  to  $v$ . We still need to argue that  $|A|, |B| \geq cn/3$  for some constant  $c$ . To lower bound  $|B|$  we introduce the event  $BAD_S = |\{i : BAD_{i,S}^{upper}\}| > \frac{k}{10}$  and note that unless  $BAD_S$  occurs we have  $|B| \geq (9k/10)n^{2/3}/2 = 9cn/40$ . Finally, we show that  $\mathbb{P}[BAD_S]$  is very small and then use union bounds to show that, for a suitable constant  $\tau$ , the probability  $\mathbb{P}[\exists SBAD_S]$  becomes negligibly small. A symmetric argument can be used to show that  $|A| \geq 9cn/40$ . ◀

We now use  $G_n$  to construct  $c$ -maximally ST-robust graphs with linear size.

► **Construction 15** ( $M_n$ ). We construct the family of graphs  $M_n$  as follows: Let the graphs  $SC_n^1$  and  $SC_n^2$  be linear sized  $n$ -superconcentrators with constant indegree  $\delta_{SC}$  [13], and let  $H_n^1, H_n^2$  and  $H_n^3$  be defined and connected as in  $G_n$ , where every output of  $SC_n^1$  is connected to a node in  $H_n^1$ , every node of  $H_n^3$  is connected to an input of  $SC_n^2$ .



■ **Figure 2** A diagram of the constant indegree, linear sized, ST-robust graph  $M_n$ .

► **Theorem 16.** *There exists a constant  $c' > 0$  such that for all sets  $S \subset V(M_n)$  with  $|S| \leq c'n/2$ ,  $M_n$  is  $(|S|, n - |S|)$ -ST-robust, with  $n$  inputs and  $n$  outputs and constant indegree.*

**Proof.** Let  $c' = 9c/40$ , where  $c$  is the constant from  $G_n$ . Consider  $M_n - S$ . Then because  $|S \cap (H_n^1 \cup H_n^2)| \leq |S| \leq c'n/2 \leq cn/2$ , by Lemma 14 with a high probability there exists sets  $A$  in  $H_n^1$  and  $B$  in  $H_n^3$  with  $|A|, |B| \geq \frac{9}{10}k\frac{n^{2/3}}{2} = \frac{9}{40}cn = c'n$ , such that every node in  $A$  connects to every node in  $B$ . By the properties of superconcentrators, the size of the set  $BAD_1$  of inputs  $u$  in  $SC_n^1$  that can't reach *any* node in  $A$  in  $M_n - S$ . We claim that  $|BAD_1| \leq |S| \leq c'n$ . Assume for contradiction that  $|BAD_1| > |S|$  then  $SC_n^1$  contains at least  $\min\{|BAD_1|, |A|\} > |S|$  node disjoint paths between  $BAD_1$  and  $A$ . At least one of these node disjoint paths does not intersect  $S$  which contradicts the definition of  $BAD_1$ . Similarly, we can bound the size of  $BAD_2$ , the set of outputs in  $SC_n^2$  which are not reachable from any node in  $B$ . Given any input  $u \notin BAD_1$  of  $SC_n^1$  and any output  $v \notin BAD_2$  of  $SC_n^2$  we can argue that  $u$  is connected to  $v$  in  $M_n - S$  since we can reach some node  $x \in A$  from  $u$  and  $v$  can be reached from some node  $y \in B$  and *any* such pair  $x, y$  is connected by a path in  $M_n - S$ . It follows that  $M_n$  is  $(|S|, n - |S|)$ -ST-robust. ◀

► **Corollary 17** (of Theorem 16). *For all  $\epsilon > 0$ , there exists a family of DAGs  $\mathbb{M} = \{M_n^\epsilon\}_{n=1}^\infty$ , where each  $M_n^\epsilon$  is a  $c$ -maximally ST-robust graphs with  $|V(M_n)| \leq c_\epsilon n$ ,  $\text{indegree}(M_n) \leq c_\epsilon$ , and depth  $d = n^{1-\epsilon}$ .*

**Proof (sketch).** In the proof of Lemma 13, we used  $(cn, n^{2/3})$ -depth robust graphs. When considering the paths  $A_i$  and  $B_j$ , we were considering connecting the upper half of one path to the lower half of another. Thus, after we remove nodes from  $M_n$ , there exists a path of



length at least  $n^{2/3}$  that connects any remaining input to any remaining output. Thus  $M_n$  is  $c$ -maximally ST-robust with depth  $d = n^{2/3}$ . In [14], Schnitger provides a construction that is  $(cn, n^{1-\epsilon})$ -depth robust for all constant  $\epsilon > 0$ . By the same arguments we used in this section, we can construct  $c$ -maximally ST-robust graphs with depth  $d = n^{1-\epsilon}$ , where the constant  $c$  depends on  $\epsilon$ .  $\blacktriangleleft$

### 3.3 Constructing Maximal ST-Robust Graphs

In this section, we construct maximal ST-robust graphs, which are 1-maximally ST-robust, from  $c$ -maximally ST-robust graphs. We give the following construction:

► **Construction 18** ( $\mathbb{O}(M_n)$ ). Let  $M_n$  be a  $c$ -maximally ST-robust graph on  $O(n)$  nodes. Let  $O$  be a set  $o_1, o_2, \dots, o_n$  of  $n$  output nodes and let  $I$  be a set  $i_1, i_2, \dots, i_n$  of  $n$  input nodes. Let  $S_j$  for  $1 \leq j \leq \lceil \frac{1}{c} \rceil$  be a copy of  $M_n$  with outputs  $o_1^j, o_2^j, \dots, o_n^j$  and inputs  $i_1^j, i_2^j, \dots, i_n^j$ . Then for all  $1 \leq j \leq n$  and for all  $1 \leq k \leq n$ , add a directed edge from  $i_k$  to  $i_k^j$  and from  $o_k^j$  to  $o_k$ .

Because we connect  $\lceil \frac{1}{c} \rceil$  copies of  $M_n$  to the output nodes,  $\mathbb{O}(M_n)$  has indegree  $\max\{\delta, \lceil \frac{1}{c} \rceil\}$ , where  $\delta$  is the indegree of  $M_n$ . Also, if  $M_n$  has  $kn$  nodes, then  $\mathbb{O}(M_n)$  has  $(k\lceil \frac{1}{c} \rceil + 2)n$  nodes. We now show that  $\mathbb{O}(M_n)$  is a maximal ST-robust graph.

► **Theorem 19.** Let  $M_n$  be a  $c$ -maximally ST-robust graph. Then  $\mathbb{O}(M_n)$  is a maximal ST-robust graph.

**Proof.** Let  $R \subset V(\mathbb{O}(M_n))$  with  $|R| = k$ . Let  $R = R_I \cup R_M \cup R_O$ , where  $R_I = R \cap I$ ,  $R_O = R \cap O$ , and  $R_M = R \cap \left(\bigcup_{i=1}^{\lceil 1/c \rceil} S_i\right)$ . Consider  $\mathbb{O}(M_n) - R$ . We see that  $|R_M| \leq k$ , so by the Pidgeonhole Principal at least one  $S_j$  has less than  $cn$  nodes removed, say it has  $t$  nodes removed for  $t \leq cn$ . Hence  $t \leq |R_M|$ . Since  $S_j$  is  $c$ -max ST-robust there exists a subgraph  $H$  of  $S_j - R$  containing  $n - t$  inputs and  $n - t$  outputs such that every input is connected to all of the outputs. Let  $H'$  be the subgraph induced by the nodes in  $V(H) \cup I' \cup O'$ , where  $I' = \{(i_a, i_a^b) | i_a^b \in H\}$  and  $O' = \{(o_a^b, o_a) | o_a^b \in H\}$ .

▷ **Claim 20.** The graph  $H'$  contains at least  $n - k$  inputs and  $n - k$  outputs and there is a path between every pair of input and output nodes.

**Proof.** The set  $|I \setminus I'| \leq |I \cap R| + |V(S_j) \cap R| \leq |R| \leq k$ . Similarly,  $|O \setminus O'| \leq |O \cap R| + |V(S_j) \cap R| \leq |R| \leq k$ . Let  $v \in I'$  be some input. By the connectivity of  $H$ ,  $v$  can reach all of the outputs in  $O'$ . Thus there is a path between every pair of input and output nodes.  $\blacktriangleleft$

Thus  $\mathbb{O}(M_n)$  is  $(k, n - k)$ -ST-robust for all  $1 \leq k \leq n$ . Therefore  $\mathbb{O}(M_n)$  is a maximal ST-robust graph.  $\blacktriangleleft$

► **Corollary 21** (of Theorem 19). For all  $\epsilon > 0$ , there exists a family  $\mathbb{M}^\epsilon = \{M_k^\epsilon\}_{k=1}^\infty$  of max ST-robust graphs of depth  $d = n^{1-\epsilon}$  such that  $|V(M_k^\epsilon)| \leq c_\epsilon n$  and  $\text{indegree}(M_k^\epsilon) \leq c_\epsilon$ .

**Proof.** Apply Construction 18 to the family graphs  $\mathbb{M}^\epsilon = \{M_k^\epsilon\}_{k=1}^\infty$  from Corollary 17. Then by Theorem 19, the family of graphs  $\{\mathbb{O}(M_k^\epsilon)\}_{k=1}^\infty$  is the desired family.  $\blacktriangleleft$

## 4 Applications of ST-Robust Graphs

As outlined previously maximally ST-Robust graphs give us a tight connection between edge-depth robustness and node-depth robustness. Because edge-depth-robust graphs are often easier to design than node-depth robust graphs [14] this gives us a fundamentally new approach to construct node-depth-robust graphs. Beyond this exciting connection we can also use ST-robust graphs to construct perfectly tight proofs of space [9, 12] and asymptotically superior wide-block labeling functions [6].

### 4.1 Tight Proofs of Space

In Proof of Space constructions [12] we want to find a DAG  $G = (V, E)$  with small indegree along with a challenge set  $V_C \subseteq V$ . Intuitively, the prover will label the graph  $G$  using a hash function  $H$  (often modeled as a random oracle in security proofs) such that a node  $v$  with parents  $v_1, \dots, v_\delta$  is assigned the label  $L_v = H(L_{v_1}, \dots, L_{v_\delta})$ . The prover commits to storing  $L_v$  for each node  $v$  in the challenge set  $V_C$ . The pair  $(G, V_C)$  is said to be  $(s, t, \epsilon)$ -hard if for any subset  $S \subseteq V$  of size  $|S| \leq s$  at least  $(1 - \epsilon)$  fraction of the nodes in  $V_C$  have depth  $\geq t$  in  $G - S$  – a node  $v$  has depth  $\geq t$  in  $G - S$  if there is a path of length  $\geq t$  ending at node  $v$ . Intuitively, this means that if a cheating prover only stores  $s \leq |V_C|$  labels and is challenged to reproduce a random label  $L_v$  with  $v \in V_C$  that, except with probability  $\epsilon$ , the prover will need at least  $t$  sequential computations to recover  $L_v$  – as long as  $t$  is sufficiently large the verifier the cheating prover will be caught as he will not be able to recover the label  $L_v$  in a timely fashion. Pietrzak argued that  $(s, t, \epsilon)$ -hard graphs translate to secure Proofs of Space in the parallel random oracle model [12].

We want  $G$  to have small indegree  $\delta(G)$  (preferably constant) as the prover will need  $O(N\delta(G))$  steps and we want  $|V_C| = \Omega(N)$  and we want  $\epsilon$  to be small while  $s, t$  should be larger. Pietrzak [12] proposed to let  $G_\epsilon$  be an  $\epsilon$ -extreme depth-robust graph with  $N' = 4N$  nodes and to let  $V_C = [3N + 1, 4N]$  be the last  $N$  nodes in this graph. An  $\epsilon$ -extreme depth-robust graph with  $N'$  nodes is  $(e, d)$ -depth robust for any  $e + d \leq (1 - \epsilon)N'$ . Such a graph is  $(s, N, s/N + 4\epsilon)$ -hard for any  $s \leq N$ . Alwen et al. [4] constructed  $\epsilon$ -extreme depth-robust graphs with indegree  $\delta(G) = O(\log N)$  though the hidden constants seem to be quite large. Thus, it would take time  $O(N \log N)$  for the prover to label the graph  $G$ . We remark that  $\epsilon = s/|V_C|$  is the tightest possible bound one can hope for as the prover can always store  $s$  labels from the set  $V_C$ .

We remark that if we take  $V_C$  to be any subset of output nodes from a maximally ST-robust graph and overlay and  $(e = s, d = t)$ -depth robust graph over the input nodes then the resulting graph will be  $(s, t, \epsilon = s/|V_C|)$ -hard – optimally tight in  $\epsilon$ . In particular, given a DAG  $G = (V = [N], E)$  with  $N$  nodes devise the overlay graph  $H_G$  by starting with a maximally ST-Robust graph with  $|V|$  inputs  $I = \{x_1, \dots, x_{|V|}\}$  and  $|V|$  outputs  $O$  then for every directed edge  $(u, v) \in E(G)$  we add the directed edge  $(x_u, x_v)$  to  $E(H_G)$  and we specify a target set  $V_C \subseteq O$ . Fisch [9] gave a practical construction of  $(G, V_C)$  with indegree  $O(\log N)$  that is  $(s, N, \epsilon = s/N + \epsilon')$ -hard. The constant  $\epsilon'$  can be arbitrarily small though the number of nodes in the graph scales with  $O(N \log 1/\epsilon')$ . Utilizing ST-robust graphs we fix  $\epsilon' = 0$  without increasing the size of the graph<sup>1</sup>.

<sup>1</sup> As a disclaimer we are not claiming that our construction would be more efficient than [9] for practical parameter settings.

► **Theorem 22.** *If  $G$  is  $(e, d)$ -depth robust then the pair  $(H_G, V_C)$  specified above is  $(s, t = d + 1, s/|V_C|)$ -hard for any  $s \leq e$ .*

**Proof.** Let  $S$  be a subset of  $|S| \leq s$  nodes in  $H_G$ . By maximal ST-robustness we can find a set  $A$  of  $N - |S|$  inputs and  $B$  of  $N - |S|$  outputs such that every pair of nodes  $u \in A$  and  $v \in B$  are connected in  $H_G - S$ . We also note since  $A$  contains all but  $s$  nodes from  $G$  that some node  $u \in A$  is the endpoint of a path of length  $t$  by  $(s, t)$ -depth-robustness of  $G$ . Since  $u$  is connected to *every* node in  $B$  this means that every node  $v \in B$  is the endpoint of a path of length *at least*  $t + 1$ . ◀

This result immediately leads to a  $(s, N^{1-\epsilon}, s/N)$ -hard pair for any  $s \leq N$  which the prover can label in  $O(N)$  time as the DAG  $G$  has constant indegree. We expect that in many settings  $t = N^{1-\epsilon}$  would be sufficiently large to ensure that a cheating prover is caught with probability  $s/N$  after each challenge i.e., if the verifier expects a response within 3 seconds, but it would take longer to evaluate the hash function  $H$   $N^{1-\epsilon}$  sequential times.

► **Corollary 23.** *For any constant  $\epsilon > 0$  there is a constant indegree DAG  $G$  with  $O(N)$  nodes along with a target set  $V_C \subseteq V(G)$  of  $N$  nodes such that the pair  $(G, V_C)$  is  $(s, t = N^{1-\epsilon}, s/N)$ -hard for any  $s \leq N$ .*

**Proof (sketch).** Let  $G$  be an  $(N, N^{1-\epsilon})$ -depth robust graph with  $N' = O(N)$  nodes and constant indegree from [14]. We can then take  $V_C$  to be any subset of  $N$  output nodes in the graph  $H_G$  and apply Theorem 22. ◀

If one does not want to relax the requirement that  $t = \Omega(N)$  then we can provide a perfectly tight construction with  $O(N \log N)$  nodes and constant indegree. Since the graph has constant indegree it will take  $O(N \log N)$  work for the prover to label the graph. This is equivalent to [12], but with perfect tightness  $\epsilon = s/N$ .

► **Corollary 24.** *For any constant  $\epsilon > 0$  there is a constant indegree DAG  $G$  with  $N' = O(N \log N)$  nodes along with a target set  $V_C \subseteq V(G)$  of  $N$  nodes such that the pair  $(G, V_C)$  is  $(s, t, s/N)$ -hard for any  $s \leq N$ .*

**Proof (sketch).** Let  $G$  be an  $(N, N \log N)$ -depth robust graph with  $N' = O(N \log N)$  nodes and constant indegree from [2]. We can then take  $V_C$  to be any subset of  $N$  output nodes in the graph  $H_G$  and apply Theorem 22. ◀

Finally, if we want to ensure that the graph only has  $O(N)$  nodes and  $t = \Omega(N)$  we can obtain a perfectly tight construction with indegree  $\delta(G) = O(\log N)$ .

► **Corollary 25.** *For any constant  $\epsilon > 0$  there is a DAG  $G$  with  $O(N)$  nodes and indegree  $\delta(G) = O(\log N)$  along with a target set  $V_C \subseteq V(G)$  of  $N$  nodes such that the pair  $(G, V_C)$  is  $(s, N, s/N)$ -hard for any  $s \leq N$ .*

**Proof (sketch).** Let  $G$  be an  $(N, N)$ -depth robust graph with  $N' = 3N$  nodes from [4]. We can then take  $V_C$  to be any subset of  $N$  output nodes in the graph  $H_G$  and apply Theorem 22. ◀

## 4.2 Wide-Block Labeling Functions

Chen and Tessaro [6] introduced source-to-sink depth robust graphs as a generic way of obtaining a wide-block labeling function  $H_{\delta, W} : \{0, 1\}^{\delta W} \rightarrow \{0, 1\}^W$  from a small-block function  $H_{fix} : \{0, 1\}^{2L} \rightarrow \{0, 1\}^L$  (modeled as an ideal primitive). In their proposed

approach one transforms a graph  $G$  with indegree  $\delta$  and into a new graph  $G'$  by replacing every node with a source-to-sink depth-robust graph. Labeling a graph  $G$  with a wide-block labeling function is now equivalent to labeling  $G'$  with the original labeling function  $H_{fix}$ . The formal definition of Source-to-Sink-Depth-Robustness is presented below:

► **Definition 26** (Source-to-Sink-Depth-Robustness (SSDR) [6]). *A DAG  $G = (V, E)$  is  $(e, d)$ -source-to-sink-depth-robust (SSDR) if and only if for any  $S \subset V$  where  $|S| \leq e$ ,  $G - S$  has a path (with length at least  $d$ ) that starts from a source node of  $G$  and ends up in a sink node of  $G$ .*

If  $G$  is  $(e, d)$ -depth robust and  $G'$  is constructed by replacing every node  $v$  in  $G$  with a  $(e^*, d^*)$ -source-to-sink-depth-robust (SSDR) and orienting incoming (resp. outgoing) edges into the sources (resp. out of the sinks) then the graph  $G'$  is  $(ee^*, dd^*)$ -depth robust [6] and has cumulative pebbling complexity at least  $ed(e^*d^*)$  [3]. The SSDR graphs constructed in [6] are  $(\frac{K}{4}, \frac{\delta K^2}{2})$ -SSDR with  $O(\delta K^2)$  vertices and constant indegree. They fix  $K := W/L$  as the ratio between the length of outputs for  $H_{\delta, W} : \{0, 1\}^{\delta W} \rightarrow \{0, 1\}^W$  and the ideal primitive  $H_{fix}$ . Their graph has  $\delta K$  source nodes for a tunable parameter  $\delta \in \mathbb{N}$ ,  $O(\delta K^2)$  vertices and constant indegree. Ideally, since we are increasing the number of nodes by a factor of  $\delta K^2$  we would like to see the cumulative pebbling complexity increase by a quadratic factor of  $\delta^2 K^4$ . Instead, if we start with an  $(e, d)$ -depth robust graph with cumulative pebbling complexity  $O(ed)$  their final graph  $G'$  has cumulative pebbling complexity  $ed \times \frac{\delta K^3}{8}$ . Chen and Tessaro left the problem of finding improved source-to-sink depth-robust graphs as an open research question.

Our construction of ST-robust graphs can asymptotically<sup>2</sup> improve some of their constructions, specifically their constructions of source-to-sink-depth-robust graphs and wide-block labeling functions.

► **Theorem 27.** *Let  $G$  be a maximal ST-robust graph with depth  $d$  and  $n$  inputs and outputs. Then  $G$  is an  $(n - 1, d)$ -SSDR graph.*

**Proof.** By the maximal ST-robustness property,  $n - 1$  arbitrary nodes can be removed from  $G$  and there will still exist at least one input node that is connected to at least one output node. Since  $G$  has depth  $d$ , the path between the input node and output node must have length at least  $d$ . ◀

By applying Theorem 27 to the construction in Corollary 19, we can construct a family of  $(\delta K, (\delta K)^{1-\epsilon})$ -SSDR graphs with  $O(\delta K)$  nodes and constant indegree and  $\delta K$  sources. In this case the cumulative pebbling complexity of our construction would be already be  $ed \times \delta^2 K^{2-\epsilon}$  which is much closer to the quadratic scaling that we would ideally like to see. We are off by just  $K^\epsilon$  for a constant  $\epsilon > 0$  that can be arbitrarily small. To make the comparison easier we could also applying Theorem 27 to obtain a family of  $(\delta K^2, (\delta K^2)^{1-\epsilon})$ -SSDR graphs with  $O(\delta K^2)$ -nodes and constant indegree. While the size of the SSDR matches [6] our new graph is  $(e\delta K^2, d(\delta K^2)^{1-\epsilon})$ -depth robust and has cumulative pebbling complexity  $ed \times \delta^{2-\epsilon} K^{4-2\epsilon} \gg ed\delta K^3$ .

<sup>2</sup> While we improve the asymptotic performance we do not claim to be more efficient for practical values of  $\delta, K$ .

## References

- 1 Joël Alwen and Jeremiah Blocki. Efficiently computing data-independent memory-hard functions. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part II*, volume 9815 of *LNCS*, pages 241–271. Springer, Heidelberg, August 2016. doi:10.1007/978-3-662-53008-5\_9.
- 2 Joël Alwen, Jeremiah Blocki, and Ben Harsha. Practical graphs for optimal side-channel resistant memory-hard functions. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017*, pages 1001–1017. ACM Press, October / November 2017. doi:10.1145/3133956.3134031.
- 3 Joël Alwen, Jeremiah Blocki, and Krzysztof Pietrzak. Depth-robust graphs and their cumulative memory complexity. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part III*, volume 10212 of *LNCS*, pages 3–32. Springer, Heidelberg, April / May 2017. doi:10.1007/978-3-319-56617-7\_1.
- 4 Joël Alwen, Jeremiah Blocki, and Krzysztof Pietrzak. Sustained space complexity. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part II*, volume 10821 of *LNCS*, pages 99–130. Springer, Heidelberg, April / May 2018. doi:10.1007/978-3-319-78375-8\_4.
- 5 Jeremiah Blocki, Benjamin Harsha, Siteng Kang, Seunghoon Lee, Lu Xing, and Samson Zhou. Data-independent memory hard functions: New attacks and stronger constructions. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 573–607. Springer, Heidelberg, August 2019. doi:10.1007/978-3-030-26951-7\_20.
- 6 Binyi Chen and Stefano Tessaro. Memory-hard functions from cryptographic primitives. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 543–572. Springer, Heidelberg, August 2019. doi:10.1007/978-3-030-26951-7\_19.
- 7 Stefan Dziembowski, Sebastian Faust, Vladimir Kolmogorov, and Krzysztof Pietrzak. Proofs of space. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 585–605. Springer, Heidelberg, August 2015. doi:10.1007/978-3-662-48000-7\_29.
- 8 Paul Erdős, Ronald L. Graham, and Endre Szemerédi. On sparse graphs with dense long paths, 1975.
- 9 Ben Fisch. Tight proofs of space and replication. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part II*, volume 11477 of *LNCS*, pages 324–348. Springer, Heidelberg, May 2019. doi:10.1007/978-3-030-17656-3\_12.
- 10 Ofer Gabber and Zvi Galil. Explicit constructions of linear-sized superconcentrators. *Journal of Computer and System Sciences*, 22(3):407–420, 1981.
- 11 Mohammad Mahmoody, Tal Moran, and Salil P. Vadhan. Publicly verifiable proofs of sequential work. In Robert D. Kleinberg, editor, *ITCS 2013*, pages 373–388. ACM, January 2013. doi:10.1145/2422436.2422479.
- 12 Krzysztof Pietrzak. Proofs of catalytic space. In Avrim Blum, editor, *ITCS 2019*, volume 124, pages 59:1–59:25. LIPIcs, January 2019. doi:10.4230/LIPIcs.ITCS.2019.59.
- 13 Nicholas Pippenger. Superconcentrators. *SIAM J. Comput.*, 6(2):298–304, 1977.
- 14 Georg Schnitger. On depth-reduction and grates. In *24th Annual Symposium on Foundations of Computer Science, Tucson, Arizona, USA, 7-9 November 1983*, pages 323–328. IEEE Computer Society, 1983. doi:10.1109/SFCS.1983.38.
- 15 Claude Shannon. Memory requirements in a telephone exchange. *Bell System Technical Journal*, 29(3):343–349, July 1950.
- 16 Leslie G. Valiant. Graph-theoretic properties in computational complexity. *J. Comput. Syst. Sci.*, 13(3):278–285, December 1976. doi:10.1016/S0022-0000(76)80041-4.
- 17 Leslie G. Valiant. Graph-theoretic arguments in low-level complexity. In Jozef Gruska, editor, *Mathematical Foundations of Computer Science 1977, 6th Symposium, Tatranska Lomnica, Czechoslovakia, September 5-9, 1977, Proceedings*, volume 53 of *Lecture Notes in Computer Science*, pages 162–176. Springer, 1977. doi:10.1007/3-540-08353-7\_135.

## A

 Connector Graphs

We say that a directed acyclic graph  $G = (V, E)$  with  $n$  input vertices and  $n$  output vertices is an  **$n$ -connector** if for any *ordered list*  $x_1, \dots, x_r$  of  $r$  inputs and any *ordered list*  $y_1, \dots, y_r$  of  $r$  outputs,  $1 \leq r \leq n$ , there are  $r$  vertex-disjoint paths in  $G$  connecting input node  $x_i$  to output node  $y_i$  for each  $i \leq r$ .

### A.1 Connector Graphs are ST-Robust

We remarked in the paper that any  **$n$ -connector** is maximally ST-robust.

► **Reminder of Theorem 11.** *If  $G$  is an  $n$ -connector, then  $G$  is  $(k, n - k)$ -ST-robust, for all  $1 \leq k \leq n$ .*

**Proof of Theorem 11.** Let  $D \subseteq V(G)$  with  $|D| = k$ . Consider  $G - D$ . Let  $A = \{(s_1, t_1), \dots, (s_m, t_m)\}$ , where the input  $s_i \in S$  is disconnected from the output  $t_i \in T$  in  $G - D$ , or  $s_i \in D$  or  $t_i \in D$ . Let  $B = \emptyset$ .

Perform the following procedure on  $A$  and  $B$ : Pick any pair  $(s_p, t_p) \in A$  and add  $s_p$  and  $t_p$  to  $B$ . Then remove the pair from  $A$  along with any other pair in  $A$  that shares either  $s_p$  or  $t_p$ . Continue until  $A$  is empty.

If we consider the nodes of  $B$  in  $G$ , then there are  $|B|$  vertex-disjoint paths between the pairs in  $B$  by the connector property, and in  $G - D$  at least one vertex is removed from each path. Thus  $|B| \leq k$ , or we have a contradiction.

If  $(s, t) \in G - (D \cup B)$  are an input to output pair, and  $s$  is disconnected from  $t$ , then by the definition of  $A$  and  $B$  we would have a contradiction, since  $(s, t)$  would still be in  $A$ . Thus all of the remaining inputs in  $G - (D \cup B)$  are connected to all the remaining outputs.

Hence, if we let  $H = G - (D \cup B)$ , then  $H$  is a subgraph of  $G$  with at least  $n - k$  inputs and  $n - k$  outputs, and there is a path going from each input of  $H$  to each of its outputs. Therefore,  $G$  is  $(k, n - k)$ -ST-robust for all  $1 \leq k \leq n$ . ◀

### Butterfly Graphs

A well known family of constant indegree  $n$ -connectors, for  $n = 2^k$ , are the  $k$ -dimensional butterfly graphs  $B_k$ , which are formed by connecting two FFT graphs on  $n$  inputs back to back. By Theorem 11, the butterfly graph is also a maximally ST-robust graph. However, the butterfly graph has  $\Omega(n \log n)$  nodes and does not yield a ST-robust graph of linear size. Since  $B_k$  has  $O(n \log n)$  vertices and indegree of 2, a natural question to ask is if there exists  $n$ -connectors with  $O(n)$  vertices and constant indegree.

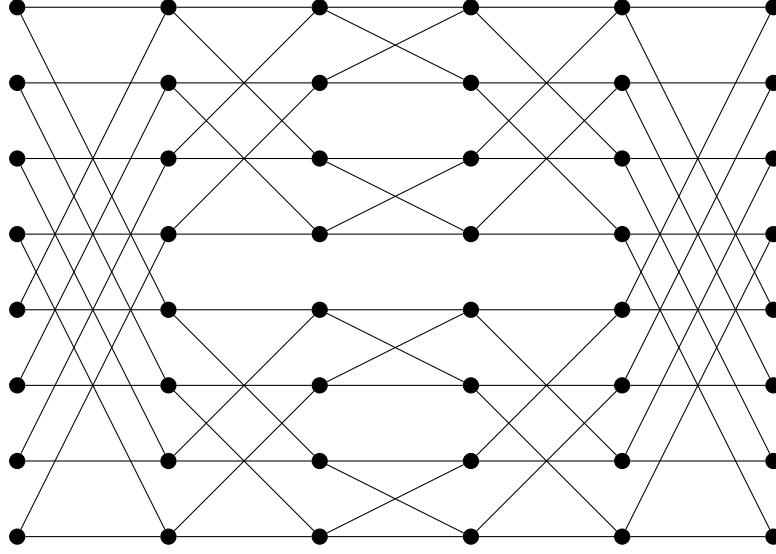
### A.2 Connector Graphs Have $\Omega(n \log n)$ vertices

An information theoretic argument of Shannon [15] rules out the possibility of linear size  $n$ -connectors.

► **Theorem 28** (Shannon [15]). *An  $n$ -connector with constant indegree requires at least  $\Omega(n \log n)$  vertices.*

Intuitively, given a  $n$ -connector with constant indegree with constant indegree and  $m$  edges Shannon argued that we can use the  $n$ -connector to encode any permutation of  $[n]$  using  $m$  bits. In more detail fixing any permutation  $\pi$  we can find  $n$  node disjoint paths from input  $i$  to output  $\pi(i)$ . Because the paths are node disjoint we can encode  $\pi$  simply





■ **Figure 3** The butterfly graph  $B_3$  is both an 8-superconcentrator and an 8-connector. All edges are directed from left to right.

by specifying the subset  $S_\pi$  of directed edges which appear in one of these node disjoint paths. We require at most  $m$  bits to encode  $S_\pi$  and from  $S_\pi$  we can reconstruct the set of node-disjoint paths and recover  $\pi$ . Thus, we must have  $m = \Theta(n \log n)$  since we require  $\log n! = \Theta(n \log n)$  bits to encode a permutation.

We stress that this information theoretic argument breaks down if the graph  $G$  is only ST-robust. We are guaranteed that  $G$  contains a path from input  $i$  to output  $\pi(i)$ , but we are not guaranteed that all of the paths are node disjoint. Thus,  $S_\pi$  is insufficient to reconstruct  $\pi$ .

## B Missing Proofs

► **Reminder of Theorem 5.** *Let  $G$  be an  $(e, d)$ -edge-depth-robust DAG with  $m$  edges. Let  $\mathbb{M}$  be a family of max ST-Robust graphs with constant indegree. Then  $G' = (V', E') = \text{Reduce}(G, \mathbb{M})$  is  $(e/2, d)$ -depth robust. Furthermore,  $G'$  has maximum indegree  $\max_{v \in V(G)} \{\text{indeg}(M_{\delta(v)})\}$ , and its number of nodes is  $\sum_{v \in V(G)} |V(M_{\delta(v)})|$  where  $\delta(v) = \max\{\text{indeg}(v), \text{outdeg}(v)\}$ .*

**Proof of Theorem 5.** We know that each graph in  $\mathbb{M}$  has constant indegree, and that each node  $v$  in  $G$  will be replaced with a graph in  $\mathbb{M}$  with indegree  $\text{indeg}(M_{\delta(v)})$ . Thus  $G'$  has maximum indegree  $\max_{v \in V(G)} \{\text{indeg}(M_{\delta(v)})\}$ . Furthermore, the metanode corresponding to the node  $v$  has size  $|M_{\delta(v)}|$ . Thus  $G'$  has  $\sum_{v \in V(G)} |M_{\delta(v)}|$  nodes.

Let  $S \subset V(G')$  be a set of nodes that we will remove from  $G'$ . For a specific node  $v \in V(G)$  we let  $S_v = S \cap (\{v\} \times V_{\delta(v)})$  denote the subset of nodes deleted from the corresponding metanode. We say that the node  $v \in V(G)$  is *irreparable* with respect to  $S$  if  $|S_v| \geq \delta(v)$ ; otherwise we say that  $v$  is *reparable*. If a node  $v$  is reparable, then because the metanodes are maximally ST-Robust we can find subsets  $I_{v,S}$  and  $O_{v,S}$  (with  $|I_{v,S}|, |O_{v,S}| \geq \delta(v) - |S_v|$ ) such that each input node  $s \in I_{v,S}$  is connected to every output node in  $O_{v,S}$ .



We say that an edge  $(u, v) \in E(G)$  is *irreparable* with respect if  $u$  or  $v$  is irreparable, or if the corresponding edge  $e' = (u', v') \in E(G')$  has  $u' \notin O_{u,S}$  or  $v' \notin I_{v,S}$ . We let  $S_{irr} \subset E(G)$  be the set of irreparable edges after we remove  $S$  from  $G$ . We begin the proof by first proving two claims.

▷ **Claim 29.** Let  $P$  be a path of length  $d$  in  $G - S_{irr}$ . Then there exists a path of length at least  $d$  in  $G' - S$ .

*Proof.* In  $G - S_{irr}$  we have removed all of the irreparable edges, so any path in the graph contains only repairable edges. By definition, if  $(u, v)$  is a repairable edge, both  $u$  and  $v$  will be repairable, and  $(u, \pi_{out,u}(v)) \in O_{u,S}$  and  $(v, \pi_{in,v}(u)) \in I_{v,S}$ . Thus the edge corresponding to  $(u, v)$  in  $G' - S$  will connect the metanodes of  $u$  and  $v$ , and  $(u, \pi_{out,u}(v))$  connects to every node in  $I_{u,S}$  and  $(v, \pi_{in,v}(u))$  connects to every node in  $O_{v,S}$ . Thus the edges in  $G' - S$  corresponding to the edges in  $P$  form a path of length at least  $d$ . ◁

▷ **Claim 30.** Let  $S_{irr} \subset E(G)$  be the set of irreparable edges with respect to the removed set  $S$ . Then

$$|S_{irr}| \leq 2|S|.$$

*Proof.* If a node  $v$  is repairable with respect to  $S$  then let  $S_{irr,v}^{in} \subseteq E(G)$  (resp.  $S_{irr,v}^{out}$ ) denote the subset of edges  $(u, v) \in E(G)$  (resp.  $(v, u) \in E(G)$ ) that are irreparable because of  $S_v$  i.e., the corresponding edge  $e' = (u', v') \in E(G')$  has  $v' \notin I_{v,S}$  (resp. the corresponding edge  $(v', u') \in E(G')$  has  $v' \notin O_{v,S}$ ). Let  $S_{irr,v} = S_{irr,v}^{in} \cup S_{irr,v}^{out}$ . Similarly, if  $v$  is irreparable we let  $S_{irr,v} = \{(u, v) : (u, v) \in E(G)\} \cup \{(v, u) : (v, u) \in E(G)\}$  denote the set of all of  $v$ 's incoming and outgoing edges. We note that  $|S_{irr}| \leq \sum_v |S_{irr,v}|$  since  $S_{irr} = \bigcup_v S_{irr,v}$  any irreparable edge must be in one of the sets  $S_{irr,v}$ . Now we claim that  $|S_{irr,v}| \leq |S_v|$  where  $S_v = S \cap (\{v\} \times V_{\delta(v)})$  denote the subset of nodes deleted from the corresponding metanode. We now observe that

$$\begin{aligned} |S_{irr,v}| &\leq |S_{irr,v}^{in}| + |S_{irr,v}^{out}| \\ &\leq (\delta(v) - |I_{v,S}|) + (\delta(v) - |O_{v,S}|) \leq 2|S_v|. \end{aligned}$$

The last inequality invokes maximal ST-robustness to show that  $\delta(v) - |O_{v,S}| \leq |S_v|$  and  $\delta(v) - |I_{v,S}| \leq |S_v|$ . If a node  $v$  is irreparable then the subsets  $I_{v,S}$  and  $O_{v,S}$  might be empty since  $\delta(v) - |S_v| \leq 0$ , but it still holds that  $\delta(v) - |O_{v,S}| \leq |S_v|$  and  $\delta(v) - |I_{v,S}| \leq |S_v|$ . Thus, we have

Thus

$$|S_{irr}| \leq \sum_v |S_{irr,v}| \leq \sum_v 2|S_v| \leq 2|S|. \quad \triangleleft$$

► **Reminder of Corollary 7.** (of Theorem 5) Suppose that there exists a family  $\mathbb{M} = \{M_k\}_{k=1}^{\infty}$  of max ST-Robust graphs with depth  $d_k$  and constant indegree. Given any  $(e, d)$ -edge-depth-robust DAG  $G$  with  $n$  nodes and maximum degree  $\delta$  we can construct a DAG  $G'$  with  $n \times |M_\delta|$  nodes and constant indegree that is  $(e/2, d \cdot d_\delta)$ -depth robust.

**Proof of Corollary 7 (sketch).** We slightly modify our reduction. Instead of replacing each node  $v \in G$  with a copy of  $M_{\delta(v)}$ , we instead replace each node with a copy of  $M_{\delta,v} := M_\delta$ , attaching the edges same way as in Construction 4. Thus the transformed graph  $G'$  has

$|V(G)| \times |M_\delta|$  nodes and constant indegree. Let  $S \subset V(G')$  be a set of nodes that we will remove from  $G'$ . By Claim 29, there exists a path  $P$  in  $G' - S$  that passes through  $d$  metanodes  $M_{\delta,v_1}, \dots, M_{\delta,v_d}$ . The only difference is that each  $M_{\delta,v_i}$  is maximally ST-robust *with depth*  $d_\delta$  meaning we can assume that the sub-path  $P_i = P \cap M_{\delta,v_i}$  through each metanode has length  $|P_i| \geq d_\delta$ . Thus, the total length of the path is at least  $\sum_i |P_i| \geq d \cdot d_\delta$ . ◀

► **Reminder of Lemma 13 [14].** *For some suitable constant  $c > 0$  any any subset  $S$  of  $cn/2$  vertices of  $G_n$  the graph  $H_n^1 - S$  contains  $k = cn^{1/3}/2$  vertex disjoint paths  $A_1, \dots, A_k$  of length  $n^{2/3}$  and  $H_n^2 - S$  contains  $k$  vertex disjoint paths  $B_1, \dots, B_k$  of the same length.*

**Proof of Lemma 13 [14].** Consider  $H_n^1 - S$ . Since  $H_n^1$  is  $(cn, n^{2/3})$ -depth-robust and  $|S| = cn/2$ , there must exist a path  $A_1 = (v_1, \dots, v_{n^{2/3}})$  in  $H_n^1 - S$ . Remove all vertices of  $A_1$  and repeat to find  $A_2, \dots$ . Then we finish with  $cn/(2n^{2/3}) = cn^{1/3}/2$  vertex disjoint paths of length  $n^{2/3}$ . We perform the same process on  $H_n^2$  to find the  $B_i$ . ◀

► **Reminder of Lemma 14.** *Let  $G_n$  be defined as in Construction 12. Then for some constants  $c > 0$ , with high probability  $G_n$  has the property that for all  $S \subset V(G_n)$  with  $|S| = cn/2$  there exists  $A \subseteq V(H_n^1)$  and  $B \subseteq V(H_n^3)$  such that for every pair of nodes  $u \in A$  and  $v \in B$  the graph  $G_n - S$  contains a path from  $u$  to  $v$  and  $|A|, |B| \geq 9cn/40$ .*

**Proof of Lemma 14.** By Lemma 13, we know that in  $G_n - S$  there exists  $k := c'n^{1/3}/2$  vertex disjoint paths  $A_1, \dots, A_k$  in  $H_n^1$  of length  $n^{2/3}$  and  $k$  vertex disjoint paths  $B_1, \dots, B_k$  in  $H_n^2$  of length  $n^{2/3}$ . Here,  $c'$  is the constant from Lemma 13. Let  $U_{j,S}^i$  be the upper half of the  $j$ -th path in  $H_n^i$  and  $L_{j,S}^i$  be the lower half, both of which are relative to the removed set  $S$ .

Now for each  $i \leq k$  define the event  $BAD_{i,S}$  to be the event that there exists  $j \leq k$  s.t.,  $U_{j,S}^1$  is disconnected from  $L_{j,S}^2$ . We now set  $GOOD_S = [k] \setminus \{i : BAD_{i,S}\}$  and define

$$B_S := \bigcup_{i \in GOOD_S}^k U_{i,S}^2, \quad \text{and} \quad A_S := \bigcup_{i=1}^k L_{i,S}^1.$$

Now we claim that for every node  $u \in A_S$  and  $v \in B_S$  the graph  $G_n - S$  contains a path from  $u$  to  $v$ . Since  $u \in A_S$  we have  $u \in L_{i,S}^1$  for some  $i \leq k$ . Thus, all nodes in  $U_{i,S}^1$  are reachable from  $u$ . Since,  $v \in B_S$  we have  $v \in U_{j,S}^2$  for some good  $j \in GOOD$ . We know that  $v$  is reachable from any node in  $L_{j,S}^2$ . By definition of  $GOOD_S$  there must be an edge  $(x, y)$  from some node  $x \in U_{i,S}^1$  to some node  $y \in L_{j,S}^2$  since we already know that there is a directed path from  $u$  to  $x$  and from  $y$  to  $v$  there is a directed path from  $u$  to  $v$ . Thus, every pair of nodes in  $A_S$  and  $B_S$  are connected.

We have  $|A_S| \geq kn^{2/3} = c'n/2$ . It remains to argue that for any set  $S$  the resulting set  $|B_S| = |GOOD_S|n^{2/3}$  is sufficiently large. Now we define the event

$$BAD_S := |\{i : BAD_{i,S}\}| > \frac{k}{10}.$$

Intuitively,  $BAD_S$  occurs when more than a small fraction of the events  $BAD_{i,S}$  occur. Assuming that  $BAD_S$  never occurs then for any set  $S$  we have

$$|B_S| \geq |GOOD_S|n^{2/3} \geq (9/10)kn^{2/3} = 9c'n/20.$$

Consider, for the sake of finding the probabilities, that  $S$  is fixed before all of the random edges are added to  $G_n$ . We will then union bound over all choices of sets  $S$ . First we consider the probability that a single upper path, say  $U_{1,S}^1$  is disconnected from a particular lower

path, say  $L_{1,S}^2$ . There are  $n^{1/3}$  possible lower parts to connect to, and there are  $n^{2/3}/2$  nodes in the upper part that can connect to the lower part, and there are  $\tau$  random edges added from each node in the upper part, so we have that

$$\mathbb{P}[U_{1,S}^1 \text{ disconnected from } L_{1,S}^2] \leq \left(1 - \frac{1}{2n^{1/3}}\right)^{\tau n^{2/3}/2} \leq \left(\frac{1}{e}\right)^{\tau n^{1/3}/4}.$$

Union bounding over all indices  $j$  we have

$$\mathbb{P}[BAD_{i,S}] \leq k \left(\frac{1}{e}\right)^{\tau n^{1/3}/4} = \left(\frac{1}{e}\right)^{\tau n^{1/3}/4 - \ln k}.$$

We remark that for  $i \neq j$  the event  $BAD_{i,S}$  is independent of  $BAD_{j,S}$  since the  $\tau$  random incoming edges connected to  $L_i^2$  are sampled independently of the edges for  $L_j^2$ .

We will show that the probability of the event  $BAD_S$  is very small and then take a union bound over all possible  $S$  to show our desired result.

$$\begin{aligned} \mathbb{P}[BAD_S] &\leq \binom{k}{k/10} \mathbb{P}[BAD_{1,S} \wedge \dots \wedge BAD_{k/10,S}] \\ &= \binom{k}{k/10} \mathbb{P}[BAD_{1,S}^{upper}]^{k/10} \\ &\leq \binom{k}{k/10} \left[ \left(\frac{1}{e}\right)^{\tau n^{1/3}/4 - \ln k} \right]^{k/10} \\ &= \binom{k}{k/10} \left(\frac{1}{e}\right)^{(k\tau n^{1/3} - 4k \ln k)/40}. \end{aligned}$$

Finally, we take the union bound over every possible  $S$  of size  $cn/2$  nodes. Since  $G_n$  has  $2n$  nodes there are at most  $2^{2n} = e^{2n \ln 2}$  such sets. Thus,

$$\mathbb{P}[\exists S \text{ s.t. } BAD_S] \leq e^{2n \ln 2} \mathbb{P}[BAD_S^{upper}] \leq \left(\frac{1}{e}\right)^{(k\tau n^{1/3} - 4k \ln k)/40 - 2n \ln 2}.$$

By selecting a sufficiently large constant like  $\tau = 800/c'$  we can ensure that  $(k\tau n^{1/3} - 4k \ln k)/40 - 2n \ln 2 = 20n - 2n \ln 2 - (k \ln k)/10 \geq n$  so that

$$\mathbb{P}[\exists S \text{ s.t. } BAD_S] \leq 2^{-n}.$$

Thus, except with negligible probability for any  $S$  of size  $cn/2$  the event  $BAD_S$  does not occur for any set  $S$  selected *after*  $G_n$  is sampled.  $\blacktriangleleft$

# Towards Local Testability for Quantum Coding

Anthony Leverrier 

Inria, Paris, France

anthony.leverrier@inria.fr

Vivien Londe

Microsoft, Issy-les-moulineaux, France

<https://vivienlonde.github.io/>

vivien.londe@microsoft.com

Gilles Zémor 

Institut de Mathématiques de Bordeaux, UMR 5251, France

zemor@math.u-bordeaux.fr

---

## Abstract

We introduce the *hemicubic codes*, a family of quantum codes obtained by associating qubits with the  $p$ -faces of the  $n$ -cube (for  $n > p$ ) and stabilizer constraints with faces of dimension  $(p \pm 1)$ . The quantum code obtained by identifying antipodal faces of the resulting complex encodes one logical qubit into  $N = 2^{n-p-1} \binom{n}{p}$  physical qubits and displays local testability with a soundness of  $\Omega(1/\log(N))$  beating the current state-of-the-art of  $1/\log^2(N)$  due to Hastings. We exploit this local testability to devise an efficient decoding algorithm that corrects arbitrary errors of size less than the minimum distance, up to polylog factors.

We then extend this code family by considering the quotient of the  $n$ -cube by arbitrary linear classical codes of length  $n$ . We establish the parameters of these *generalized hemicubic codes*. Interestingly, if the soundness of the hemicubic code could be shown to be constant, similarly to the ordinary  $n$ -cube, then the generalized hemicubic codes could yield quantum locally testable codes of length not exceeding an exponential or even polynomial function of the code dimension.

**2012 ACM Subject Classification** Theory of computation → Error-correcting codes

**Keywords and phrases** Quantum error correcting code

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.65

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/1911.03069>.

**Funding** *Anthony Leverrier*: Agence Nationale de la Recherche (ANR): QuantERA project QCDA.  
*Gilles Zémor*: Agence Nationale de la Recherche (ANR): ANR-18-CE47-0010 project QUDATA.

**Acknowledgements** We would like to thank Benjamin Audoux, Alain Couvreur, Omar Fawzi, Antoine Grossepieller and Jean-Pierre Tillich for many fruitful discussions on quantum codes.

## 1 Quantum LDPC codes, local testability and robustness of entanglement

Entanglement is arguably the central concept of quantum theory and despite decades of study, many questions about it remain unsolved today. One particular mystery is the robustness of phases of highly entangled states, such as the ones involved in quantum computation. Given such a state, does it remain entangled in the presence of noise? A closely related question concerns low-energy states of local Hamiltonians: while ground states, *i.e.*, states of minimal energy, are often highly entangled, is it also the case of higher energy states? These questions are related through the concept of quantum error correction: logical information is often encoded in a quantum error correcting code (QECC) in order to be processed during a quantum computation, and the ground space of a local Hamiltonian is nothing but a special case of a QECC called quantum low-density parity-check (LDPC) code.



© Anthony Leverrier, Vivien Londe, and Gilles Zémor;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 65; pp. 65:1–65:11

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Physically it indeed makes sense to implement quantum error correction by relying on local interaction, for example by encoding the quantum state in the degenerate ground space of a local Hamiltonian, that is an  $N$ -qubit operator  $H \propto \sum_i \Pi_i$ , where each  $\Pi_i$  is a projector acting nontrivially on a small number  $q$  of qubits (we talk of  $q$ -local terms). By “small”, one usually means constant or sometimes logarithmic in  $N$ . A quantum stabilizer code is a subspace of the space  $(\mathbb{C}^2)^{\otimes N}$  of  $N$  qubits defined as the common  $+1$  eigenspace of a set  $\{S_1, \dots, S_m\}$  of commuting Pauli operators, that is, the space

$$\text{span}\{|\psi\rangle \in (\mathbb{C}^2)^{\otimes N} : S_i|\psi\rangle = |\psi\rangle, \forall i \in [m]\}.$$

Such a code is said to be *LDPC* if all the generators  $S_i$  act nontrivially on at most  $q$  qubits for small  $q$ . With this language, a quantum LDPC stabilizer code corresponds to the ground space of the local Hamiltonian  $H = \frac{1}{m} \sum_{i=1}^m \Pi_i$ , with  $\Pi_i = \frac{1}{2}(I - S_i)$ .

Entanglement can be quantified in many ways, but a relevant definition is to say that a quantum state is highly entangled (or displays *long-range entanglement*) if it cannot be obtained by processing an initial product state via a quantum circuit of constant depth. By contrast, a quantum state that can be obtained that way, and which is therefore of the form  $U_{\text{circ}}(\otimes_{i=1}^n |\phi_i\rangle)$  for some  $|\phi_i\rangle \in \mathbb{C}^2$ , is said to be *trivial*. An important property of trivial states is that they admit an efficient classical description and that one can efficiently compute the value of local observables such as  $\Pi_i$  for such states: this is because the operator  $U_{\text{circ}}^\dagger \Pi_i U_{\text{circ}}$  remains local (since the circuit has constant depth) and its expectation can therefore be computed efficiently for a product state. In particular, such a classical description can serve as a witness that a local Hamiltonian admits a trivial state of low energy. It is well known how to construct  $N$ -qubit Hamiltonians with highly entangled ground states, for instance by considering a Hamiltonian associated with a quantum LDPC code with non-constant minimum distance [9], but the question of the existence of local Hamiltonians such that low-energy states are non-trivial remains poorly understood.

The no low-energy trivial state (NLTS) conjecture asks whether there exists a local Hamiltonian such that all states of small enough (normalized) energy are nontrivial [20]. More precisely, is there some  $H = \frac{1}{m} \sum_{i=1}^m \Pi_i$  as above, such that there exists a constant  $\alpha > 0$  such that all states  $\rho$  satisfying  $\text{tr}(\rho H) \leq \alpha$  are nontrivial? What is interesting with the NLTS conjecture is that it is a consequence of the quantum PCP conjecture [1], and therefore corresponds to a possible milestone on the route towards establishing the quantum PCP conjecture. We note that there are several versions of the quantum PCP conjecture in the literature, corresponding to the quantum generalizations of equivalent versions of the classical PCP theorem, but not known to be equivalent in the quantum case, and that the multiprover version was recently established [28]. Here, however, we are concerned with the Hamiltonian version of the quantum PCP conjecture which still remains wide open. This conjecture is concerned with the complexity of the *Local Hamiltonian* problem: given a local Hamiltonian as before, two numbers  $a < b$  and the promise that the minimum eigenvalue of the Hamiltonian is either less than  $a$ , or greater than  $b$ , decide which is the case. The quantum PCP conjecture asserts that this problem is QMA-hard when the gap  $b - a$  is constant. This generalizes the PCP theorem that says that the satisfiability problem is NP-hard when the relative gap is constant [11]. Here, QMA is the class of languages generalizing NP (more precisely generalizing MA), where the witness can be a quantum state and the verifier is allowed to use a quantum computer. Assuming that  $\text{NP} \not\subseteq \text{QMA}$ , we see that Hamiltonians with trivial states of low energy cannot be used to prove the quantum PCP conjecture since the classical description of such states would be a witness that could be checked efficiently by a classical verifier. In other words, if the quantum PCP conjecture is true, it implies that NLTS holds. The converse statement is unknown.

Eldar and Harrow made progress towards the NLTS conjecture by establishing a simpler variant, called NLETS [15], by giving an explicit local Hamiltonian where states close to ground states are shown to be nontrivial. (See also Ref. [29] for an alternate proof exploiting approximate low-weight check codes.) The subtlety here is that closeness is not defined as “low energy” as in NLTS, but by the existence of a low weight operator mapping the state to a ground state. Viewing the ground space as a quantum LDPC code, [15] shows that states which are  $\delta N$ -close to the code (for some sufficiently small  $\delta > 0$ ) are nontrivial. The NLTS conjecture asks for something stronger: that all states with energy less than a small, constant, fraction of the operator norm of the Hamiltonian are nontrivial. Of course, states close to the codespace have a low (normalized) energy or syndrome weight, but the converse does not hold in general, and this is what makes the NLTS conjecture difficult to tackle.

One case where the distance to the code is tightly related to the syndrome weight is for *locally testable codes* (LTC): classical locally testable codes are codes for which one can efficiently decide, with high probability, whether a given word belongs to the code or is far from it, where efficiency is quantified in the number of queries to the coordinates of the word. To see the link between the two notions, the idea is to distinguish between codewords and words far from the code by computing a few elements of the syndrome and deciding that the word belongs to the code if all these elements are zero. An LTC is such that any word at constant relative distance from the code will have a constant fraction of unsatisfied checknodes, that is a syndrome of weight linear in the blocklength. The Hadamard code which maps a  $k$ -bit word  $x$  to a string of length  $2^k$  corresponding to the evaluations at  $x$  of all linear functions provides such an example with the syndrome corresponding to all possible linearity tests between the bits of the word: indeed, any word that satisfies most linearity tests can be shown to be close to the codespace [6].

While LTCs have been extensively studied in the classical literature [19] and provide a crucial ingredient for the proof of the classical PCP theorem, their quantum generalization is relatively new and much less understood. The concept was only recently introduced in a paper by Aharonov and Eldar [2] which showed that the classical approaches to local testability seem to fail in the quantum world: for instance, defining a code on a (hyper)graph with too much expansion seems to be a bad idea. In any case, if quantum LTCs with constant minimum distance existed, they would provide a proof of the NLTS conjecture [15], and this motivates trying to understand whether such codes can exist. Let us, however, mention that while classical LTCs are useful for performing alphabet reduction in the context of the PCP theorem, the same doesn’t seem to apply in the quantum regime since it is known that directly quantizing Dinur’s combinatorial proof of the PCP theorem [11] is bound to fail [8, 1].

An additional difficulty in the quantum case is that good quantum LDPC codes are not even known to exist. While taking a random LDPC code yields a code with linear minimum distance with high probability in the classical case, the same statement is not known to hold in the quantum setting. Even restricting our attention to codes only encoding a constant number of logical qubits, it is hard to find families of codes with minimum distance much larger than  $\sqrt{N}$ : a construction due to Freedman, Meyer and Luo gives a minimum distance  $\Theta(N^{1/2} \log^{1/4} N)$  [18] while recent constructions based on high-dimensional expanders yield a polylogarithmic improvement [23, 16, 24] and hold the current record for quantum LDPC codes. (Note that considering subsystem codes [30] or approximate codes [10, 5] is helpful to get a large minimum distance [4, 29, 7].) For these reasons, while a lot of work on classical LTCs focusses on codes with linear minimum distance and aims at minimizing the length of the code, the current goals in the quantum case are much more modest at this point.

A possible formal definition of a quantum LTC was suggested by [15], which we detail now. Recall that the objective is to relate two notions: the distance of a state to the code, and the energy of the state. A quantum code, or equivalently, its associated Hamiltonian, will be locally testable if any word at distance  $t$  from the code (or the ground space) has energy  $\Omega(t)$  and if this energy can be estimated by accessing only a small number of qubits (this is why we insist on having local terms in the Hamiltonian). First, one defines a quantum version of the Hamming distance as follows. Consider the code space  $\mathcal{C} \subset (\mathbb{C}^2)^{\otimes N}$  and define its  $t$ -fattening  $\mathcal{C}_t$  as the span of states at distance at most  $t$  from  $\mathcal{C}$ :

$$\mathcal{C}_t := \text{Span}\{(A_1 \otimes \cdots \otimes A_n)|\psi\rangle : |\psi\rangle \in \mathcal{C}, |\{i : A_i \neq I\}| \leq t\},$$

where the  $A_i$  are single-qubit Pauli matrices. States at distance  $t$  belong to  $\mathcal{C}_t$ , but not to  $\mathcal{C}_{t-1}$ , which we formalize by considering the projector  $\Pi_{\mathcal{C}_t}$  onto  $\mathcal{C}_t$  and forming the *distance operator*

$$D_{\mathcal{C}} := \sum_t t(\Pi_{\mathcal{C}_t} - \Pi_{\mathcal{C}_{t-1}}).$$

Informally, the eigenspace of  $D_{\mathcal{C}}$  with eigenvalue  $t$  corresponds to states which are at distance  $t$  from the code. We now define the averaged normalized Hamiltonian  $H_{\mathcal{C}}$  associated with the quantum code  $\mathcal{C}$  with  $q$ -local projections  $(\Pi_1, \dots, \Pi_m)$ :

$$H_{\mathcal{C}} = \frac{1}{m} \sum_{i=1}^m \Pi_i.$$

The normalization by  $m$  ensures that  $\|H_{\mathcal{C}}\| \leq 1$ . With these notations, we say that a  $q$ -local quantum code  $\mathcal{C} \subseteq (\mathbb{C}^2)^{\otimes n}$  is an  $(s, q)$ -quantum LTC with soundness  $s \in [0, 1]$  if<sup>1</sup>

$$H_{\mathcal{C}} \succeq \frac{s}{N} D_{\mathcal{C}}, \tag{1}$$

where  $A \succeq B$  means that the operator  $A - B$  is positive semidefinite. In words, condition (1) means that any low-energy state is close to the codespace in terms of the quantum Hamming distance, and that simple energy tests allow one to distinguish codewords from states far from the code. More precisely, one can distinguish between a codeword (with energy 0) and a state at distance  $\delta N$  from the code (therefore with energy  $\geq s\delta$ ) by measuring approximately  $1/(s\delta)$  terms of the Hamiltonian. Ideally, one would want the soundness  $s$  and the locality  $q$  to be constant, so that accessing a constant number of qubits would suffice to distinguish codewords from states at distance greater than  $\delta N$  from the code, for constant  $\delta > 0$ .

Known constructions of quantum LTCs are rare. For instance, quantum expander codes yield one example of  $(s, q)$ -quantum LTCs with both  $s = O(1)$ ,  $q = O(1)$ , but with the *major caveat* that Eq. (1) doesn't hold in general, but only on the restriction of the Hilbert space consisting of states  $O(\sqrt{N})$ -close to the codespace [27]. In fact, there exist states at distance  $\Omega(\sqrt{N})$  violating only a single projection  $\Pi_i$ . This means that such codes cannot be used to establish the NLTS conjecture. By allowing the locality to be logarithmic in the number of qubits instead of constant, that is  $q = O(\log N)$ , a recent construction of Hastings [21] yields a quantum LTC with soundness  $s = O\left(\frac{1}{\log^2 N}\right)$ , without any restriction on the validity of Eq. (1). The construction is a generalization of the toric code where instead of taking the product of two 1-cycles of length  $p$ , one rather considers the product of two  $d$ -cycles of area  $p^d$  for the appropriate values of  $p = \omega(1)$  and  $d = \omega(1)$ .

<sup>1</sup> In a previous version of this manuscript, <https://arxiv.org/abs/1911.03069v1>, we were additionally normalizing the Hamiltonian by  $q$ , leading to a soundness value of  $s/q$ . We remove this extra factor here, in accordance with the literature in classical and quantum locally testable codes.



## Our results

In this work, we present a different construction of a quantum LTC which shares with Hastings' the property that it is set in a high-dimensional space with  $d = \Theta(\log N)$  and therefore a similar locality<sup>2</sup>  $q = \Theta(\log N)$ . Our code, however, achieves a slightly better soundness  $r = \Omega\left(\frac{1}{\log N}\right)$ , and in fact, we were not able to rule out that the soundness is not constant, which would be optimal. While this hemicube code only encodes a single logical qubit, we can introduce a generalized family of codes with polynomial rate. These codes are obtained starting with the chain complex associated to the  $n$ -dimensional Hamming cube, where we identify faces corresponding to the same coset of a classical code of length  $n$ . A CSS quantum code is obtained by placing qubits on the  $p$ -faces and stabilizers either on  $(p-1)$ -faces or  $(p+1)$ -faces, with constraints given by the incidence relations between the faces in the cube. While this construction is arguably quite natural, computing the parameters (dimension and minimum distance) of this code family turned out to be rather subtle, relying in nontrivial arguments from algebraic topology. The parameters of the CSS code resulting from the quotient of the cube by a linear code of parameters  $[n, k, d]$  are

$$\left\lfloor 2^{n-p-k} \binom{n}{p}, \binom{p+k-1}{p}, \min \left\{ \binom{d}{p}, 2^{n-p-k} \right\} \right\rfloor$$

when qubits are placed on  $p$ -faces for  $p \leq d-2$ . Whether these codes are also locally testable is left as an open question. In that case, these would provide the first examples of quantum LTCs of exponential or even polynomial length in the code dimension. Remember indeed that both the hemicubic and Hastings' codes have constant dimension.

## 2 Construction of the hemicubic code

We start with the simplest member of our quantum code family, corresponding to the quotient of the  $n$ -cube by the repetition code. It has been known since Kitaev [25] that one can associate a quantum CSS code with any chain complex of binary vector spaces of the form:  $C_2 \xrightarrow{\partial_2} C_1 \xrightarrow{\partial_1} C_0$ , where the boundary operators  $\partial_2$  and  $\partial_1$  satisfy  $\partial_1 \partial_2 = 0$ . One first defines two classical codes  $C_X = \ker \partial_1$  and  $C_Z = (\text{Im } \partial_2)^\perp = \ker \partial_2^T$ . These codes satisfy  $C_Z^\perp \subseteq C_X$  since  $\partial_1 \partial_2 = 0$  and the resulting quantum CSS code is the linear span of  $\left\{ \sum_{z \in C_Z^\perp} |x+z\rangle : x \in C_X \right\}$ , where  $\{|x\rangle : x \in \mathbb{F}_2^N\}$  is the canonical basis of  $(\mathbb{C}^2)^{\otimes N}$  and  $N$  is the dimension of the central space  $C_1$  of the chain complex. One obtains in this way a quantum code of length  $N$  and dimension  $\dim(C_X/C_Z^\perp) = \dim(C_X) + \dim(C_Z) - N$ . Its minimum distance is given by  $d_{\min} = \min(d_X, d_Z)$  with  $d_X = \min\{|w| : w \in C_X \setminus C_Z^\perp\}$  and  $d_Z = \min\{|w| : w \in C_Z \setminus C_X^\perp\}$ . Here,  $|w|$  stands for the Hamming weight of the word  $w$ .

Our construction relies on the  $n$ -dimensional hemicube, where a  $p$ -face is formed by a pair of antipodal  $p$ -dimensional faces of the Hamming cube  $\{0, 1\}^n$ . A  $p$ -face of the Hamming cube is a string of  $n$ -elements from  $\{0, 1, *\}$  where symbol  $*$  appears exactly  $p$  times. Let us denote by  $C_p^n$  the  $\mathbb{F}_2^n$ -vector space spanned by  $p$ -faces of the hemicube. Boundary  $\partial_p$  and coboundary  $\delta_p$  operators are obtained by extending the natural operators for the Hamming cube to the hemicube

$$\begin{aligned} \partial_p x_1 \dots x_n &:= \bigoplus_{i \text{ s.t. } x_i = *} x_1 \dots x_{i-1} 0 x_{i+1} \dots x_n \oplus x_1 \dots x_{i-1} 1 x_{i+1} \dots x_n \\ \delta_p x_1 \dots x_n &:= \bigoplus_{i \text{ s.t. } x_i \neq *} x_1 \dots x_{i-1} * x_{i+1} \dots x_n \end{aligned}$$

<sup>2</sup> We note that in both our construction and Hastings', each qubit is only involved in a logarithmic number of constraints.

and are further extended to  $p$ -chains by linearity. We reserve the notation  $+$  for the standard addition in  $\mathbb{F}_2$  and use  $\oplus$  for summing chains. The hemicubic code is then defined as the CSS code obtained from the chain complex

$$C_{p+1}^n \xrightarrow{\partial_{p+1}} C_p^n \xrightarrow{\partial_p} C_{p-1}^n.$$

Choosing  $p = \alpha n$  for  $0 < \alpha < 1$ , the resulting code will be LDPC with generators of logarithmic weight since the boundary and coboundary operators act nontrivially on  $O(n) = O(\log N)$  coordinates. The dimension of the hemicubic code corresponds to that of the homology groups  $H_p^n = \ker \partial_p / \text{Im } \partial_{p+1}$ . Since the hemicube, viewed as a cellular complex, has the same topology as the real projective plane, its homology groups all have the same dimension equal to 1. We note that the quantum code obtained here can be described with a completely different approach exploiting Khovanov homology [3]. Obtaining the minimum distance of the code requires more care since one needs to find lower bounds on the weight of minimal nontrivial cycles and cocycles in the hemicube. Summarizing, we establish the following result.

► **Theorem 1.** *The hemicubic code is a CSS code with parameters*

$$\left[ \left[ N = 2^{n-p-1} \binom{n}{p}, 1, d_{\min} = \min \left\{ \binom{n}{p}, 2^{n-p-1} \right\} \right] \right].$$

Let  $\alpha^* \approx 0.227$  be the unique nonzero solution of  $h(\alpha^*) = 1 - \alpha^*$  where  $h$  is the binary entropy function. Then choosing  $p = \lfloor \alpha^* n \rfloor$  yields a quantum code family with  $d_{\min} \geq \frac{\sqrt{N}}{1.62}$  [3].

### 3 Local testability of the hemicubic code

We now turn our attention to the local testability of the hemicubic code. This property results from isoperimetric bounds on the hemicube.

► **Theorem 2.** *The hemicubic code is locally testable with soundness  $s = \Omega\left(\frac{1}{\log N}\right)$ .*

This improves over Hastings' construction [22] obtained by taking the product of two  $n$ -spheres and which displays soundness  $s = \Theta(\log^{-2}(N))$ . It would be interesting to understand whether the bounds of Theorem 2 are tight or not. At the moment, we believe it might be possible to get rid of the logarithmic factor and obtain a constant soundness for the hemicubic code. This would then match the soundness of the standard Hamming cube, which does not encode any logical qubit since its associated complex has zero homology.

We say that a  $p$ -chain  $X$  is a *filling* of  $Y$  if  $\partial X = Y$  and that a  $p$ -cochain  $X$  is a *cofilling* of  $Y$  if  $\delta X = Y$ . The main tools to establish the soundness of the hemicubic code are upper bounds on the size of fillings (resp. cofillings) for boundaries (resp. coboundaries) in the cube. Denoting the Hamming weight of chains and cochains by  $\| \cdot \|$ , we have:

► **Lemma 3.** *Let  $E$  be a  $p$ -chain of  $C_p^n$ . Then there exists a  $p$ -chain  $F$  which is a filling of  $\partial E$ , satisfying  $\partial F = \partial E$  such that*

$$\|F\| \leq \frac{n-p}{2} \|\partial E\|.$$

*Let  $E$  be a  $p$ -cochain of  $C_p^n$ . Then there exists a  $p$ -cochain  $F$  which is a cofilling of  $\delta E$ , satisfying  $\delta F = \delta E$  such that*

$$\|F\| \leq (p+1) \|\delta E\|.$$

It is straightforward to translate these results in the language of quantum codes. Let us represent an arbitrary Pauli error of the form  $\bigotimes_{i \in E_X, j \in E_Z} X^i Z^j$  by a couple  $E = (E_X, E_Z)$  where  $E_X$  is the support of the  $X$ -type errors and  $E_Z$  is the support of the  $Z$ -type error. Interpreting  $E_X$  as a  $p$ -chain and  $E_Z$  as a  $p$ -cochain, we see that the syndrome of  $E$  is given by the pair  $(\partial E_X, \delta E_Z)$ . In order to compute the soundness of the quantum code, one needs to lower bound the ratio:

$$\min_{(E_X, E_Z)} \frac{\|\partial E_X\| + \|\delta E_Z\|}{\|[E_X]\| + \|[E_Z]\|} \geq \min \left\{ \min_{E_X} \frac{\|\partial E_X\|}{\|[E_X]\|}, \min_{E_Z} \frac{\|\delta E_Z\|}{\|[E_Z]\|} \right\},$$

where the minimum is computed over all errors with a nonzero syndrome, i.e., for  $p$ -chains  $E_X$  which are not a  $p$ -cycle and  $p$ -cochains  $E_Z$  which are not a  $p$ -cocycle. In these expressions, we denote by  $[E]$  the representative of the equivalence class of error  $E$ , with the smallest weight. Indeed, recall that two errors differing by an element of the stabilizer group are equivalent. The fact that one considers  $[E]$  instead of  $E$  makes the analysis significantly subtler in the quantum case than in the classical case. A solution is to work backward (as was also done by Dotterer in the case of the Hamming cube [13]): start with a syndrome and find a small weight error giving rise to this syndrome. This is essentially how we establish Lemma 3:

$$\min_{E_X, \partial E_X \neq 0} \frac{\|\partial E_X\|}{\|[E_X]\|} \geq \frac{2}{n-p}, \quad \min_{E_Z, \delta E_Z \neq 0} \frac{\|\delta E_Z\|}{\|[E_Z]\|} \geq \frac{1}{p+1}.$$

This implies the soundness in Theorem 2 since  $n-p, p+1 = \Theta(\log N)$ .

While Dotterer established tight bounds for the size of (co)fillings in the Hamming cube, we do not know whether the bounds of Lemma 3 are tight. Right now, we lose a logarithmic factor in the case of the hemicube, but it is not clear that this should be the case. In fact, it is not even excluded that the hemicube could display a *better* soundness than the standard cube. We expand on these ideas in the full version of the paper [26].

## 4 An efficient decoding algorithm for the hemicubic code

The existence of the small fillings and cofillings promised by the soundness of the code is particularly interesting in the context of decoding since it guarantees the existence of a low-weight error associated to any low-weight syndrome. To turn this into an efficient decoding algorithm, the main idea is to notice that one can efficiently find the required fillings and cofillings and therefore find Pauli errors giving the observed syndrome. While finding the smallest possible fillings or cofillings does not appear to be easy, finding ones satisfying the bounds of Lemma 3 can be done efficiently.

We note, however, that the decoding algorithm does not seem to perform so well against random errors of linear weight. In particular, arguments from percolation theory that would imply that errors tend to only form small clusters and that therefore it is sufficient to correct these errors (similarly to [17] for instance) will likely fail here because of the logarithmic weight of the generators. Indeed, the factor graph of the code has logarithmic degree and there does not exist a constant threshold for the error probability such that below this threshold, errors appear in clusters of size  $o(N)$ . In addition, and more importantly, our decoding algorithm is not local in the sense that it explores only the neighborhood of some violated constraints to take a local decision, and for this reason, it is not entirely clear whether the algorithm processes disconnected clusters of errors independently.

► **Theorem 4.** *The hemicubic code comes with an efficient decoding algorithm that corrects adversarial errors of weight  $O(d_{\min}/\log^2 N)$  with complexity  $O(n^4 N)$ .*

The decoding complexity is quasilinear in the error size and the algorithm can be parallelized to run in logarithmic depth. Finding a filling (or cofilling) can be done recursively by fixing one of the  $n$  coordinates and finding fillings in the projective cube of dimension  $n - 1$ . While the choice of the special coordinate is not immediately obvious if one wants to find the smallest filling, it is nevertheless possible to make a reasonably good choice efficiently by computing upper bounds on the final filling size for each possible choice of coordinate. We establish Theorem 4 in the full version of the paper [26].

## 5 Generalized hemicubic codes: quotients by arbitrary linear codes

A key remark is that identifying antipodal  $p$ -faces of the  $n$ -cube is equivalent to considering the cosets of the repetition code  $\{0^n, 1^n\}$  in the cube complex. It is therefore tempting to generalize this approach by identifying the elements of the cosets of arbitrary linear codes  $\mathcal{C}$  with parameters  $[n, k, d]$ . We form in this way a new complex where two  $p$ -faces  $x$  and  $y$  are identified if there exists a codeword  $c \in \mathcal{C}$  such that  $x = y + c$ . Recall that addition is coordinate-wise here and that  $*$  is an absorbing element.

Deriving the parameters of the quantum CSS code associated to these new complexes has been surprisingly challenging. In particular it does not seem particularly obvious that the quantum parameters, especially the minimum distance, should depend only on the parameters  $[n, k, d]$  of the classical code  $\mathcal{C}$  and not otherwise on its particular structure: it turns out indeed to be the case however. We managed to derive the quantum parameters by exhibiting explicit representatives of the  $\mathbb{F}_2$ -homology and cohomology classes, through a double induction on  $p$  and the classical code dimension  $k$ . We obtain a lower bound on the minimum homologically non-trivial cycle weight by exhibiting a set of representatives of a cohomology class all of which must be orthogonal to the cycle, and in particular intersect it. Since a non-trivial cycle meets this bound it is exact. A similar method is used to derive the minimum non-trivial cocycle weight and we obtain the following theorem.

► **Theorem 5.** *The quantum code obtained as the quotient of the  $n$ -cube by a linear code  $[n, k, d]$  admits parameters*

$$\left\lfloor 2^{n-p-k} \binom{n}{p}, \binom{p+k-1}{p}, \min \left\{ \binom{d}{p}, 2^{n-p-k} \right\} \right\rfloor$$

when qubits are placed on  $p$ -faces for  $p \leq d - 2$ .

An interesting case is  $k = 2$ , which yields a quantum code of exponential length (that is, dimension logarithmic in the code length):

$$\left\lfloor 2^{n-p-2} \binom{n}{p}, p + 1, \min \left\{ \binom{d}{p}, 2^{n-p-2} \right\} \right\rfloor.$$

We are only able to prove a lower bound on the soundness of the code (for  $X$ -errors) of  $\Omega(1/p!)$ . However, a much improved soundness would follow from the conjectured filling and cofilling constants of the original hemicubic complex: generalized hemicubic codes are therefore candidates for quantum locally testable codes of growing dimension, of which no examples are presently known.

## 6 Discussion and open questions

In this paper, we have introduced a family of quantum code constructions that live on the quotient of the  $n$ -dimensional Hamming cube by classical linear codes. Despite the apparent simplicity of the construction, it does not seem to have appeared before in the literature.

Deriving the parameters of these codes turned out to be significantly subtler than expected, and quite surprisingly, the parameters of the quantum code only depend on the parameters of the classical code and not on any additional structure. The simplest member of our quantum code family, the hemicubic code, basically inherits its local testability from the soundness of the Hamming cube, which was established by Dotterer. In our view, the fact that our code construction relies so much on the Hamming cube may be expected to yield additional advantages, through the import of other interesting properties from the cube, as well as tools from Boolean analysis.

The most pressing question is to understand whether the generalized hemicubic codes also display local testability. At the moment, we can only establish it for the simplest member of the family, which only encodes a single logical qubit. If we could show that the codes corresponding to the quotient of the Hamming cube by arbitrary linear codes of dimension  $k$  remain locally testable, then this would provide the first examples of quantum locally testable codes of exponential (if  $k > 1$ ) or polynomial (if  $k = \Omega(n)$ ) length. As we discuss in the full version of the paper [26], improving our bound on the soundness of the one-qubit hemicubic code from  $\frac{1}{\log N}$  to constant would already prove that the generalized code with  $k = 2$  remains locally testable. An indication that such an improvement might be possible comes from the 0-qubit code defined on the standard hypercube (without identifying antipodal faces) which indeed displays constant soundness [12]. More generally, the question of what parameters are achievable for quantum locally testable codes is essentially completely open at the moment.

Another intriguing question is whether the hemicubic code might help towards establishing the NLTS conjecture (albeit with a quasilocal Hamiltonian with terms of logarithmic weight) or more generally whether it is relevant for many-body physics. As mentioned earlier, any quantum LTC with linear minimum distance would yield such a proof [15]. The hemicubic code, however, is restricted by a  $O(\sqrt{N})$  minimum distance, and the argument of [15] does not directly apply anymore. This is in particular a line of research followed by Eldar which relies on the hemicubic code and which provides positive partial results [14]. We note that in the physics context of the Local Hamiltonian, it is crucial that every individual quantum system (say, qubit) is acted upon by a small number of terms. In this sense, the problem is somewhat more constrained than in the local testability case where one is typically fine if the number of qubits is much larger than the number of constraints. Our quantum codes satisfy this requirement since each qubit is only involved in a logarithmic number of local constraints.

Finally, while classical LTCs have found a number of applications in recent years, notably for constructing PCPs, it is fair to say that not much is presently known about possible applications of quantum LTCs. At the same time, local testability is a notion that makes perfect sense in the quantum regime and it seems reasonable to think that quantum LTCs might also find applications. Finding explicit families encoding a non-constant number of qubits is a natural first step.

---

## References

- 1 Dorit Aharonov, Itai Arad, and Thomas Vidick. Guest column: the quantum PCP conjecture. *ACM SIGACT news*, 44(2):47–79, 2013.
- 2 Dorit Aharonov and Lior Eldar. Quantum locally testable codes. *SIAM Journal on Computing*, 44(5):1230–1262, 2015.
- 3 Benjamin Audoux. An application of Khovanov homology to quantum codes. *Ann. Inst. Henri Poincaré Comb. Phys. Interact*, 1:185–223, 2014.
- 4 Dave Bacon, Steven T Flammia, Aram W Harrow, and Jonathan Shi. Sparse quantum codes from quantum circuits. In *Proceedings of the forty-seventh annual ACM symposium on Theory of Computing*, pages 327–334, 2015.

- 5 Cédric Bény and Ognian Oreshkov. General conditions for approximate quantum error correction and near-optimal recovery channels. *Physical review letters*, 104(12):120501, 2010.
- 6 Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-testing/correcting with applications to numerical problems. *Journal of computer and system sciences*, 47(3):549–595, 1993.
- 7 Thomas C Bohdanowicz, Elizabeth Crosson, Chinmay Nirkhe, and Henry Yuen. Good approximate quantum ldpc codes from spacetime circuit hamiltonians. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 481–490, 2019.
- 8 Fernando GSL Brandao and Aram W Harrow. Product-state approximations to quantum ground states. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 871–880. ACM, 2013.
- 9 S Bravyi, MB Hastings, and F Verstraete. Lieb-robinson bounds and the generation of correlations and topological quantum order. *Physical Review Letters*, 97(5):050401, 2006.
- 10 Claude Crépeau, Daniel Gottesman, and Adam Smith. Approximate quantum error-correcting codes and secret sharing schemes. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 285–301. Springer, 2005.
- 11 Irit Dinur. The PCP theorem by gap amplification. *Journal of the ACM (JACM)*, 54(3):12, 2007.
- 12 Dominic Dotterrer. *The (co) isoperimetric problem in (random) polyhedra*. PhD thesis, University of Toronto, 2013.
- 13 Dominic Dotterrer. The filling problem in the cube. *Discrete & Computational Geometry*, 55(2):249–262, 2016.
- 14 Lior Eldar. Robust quantum entanglement at (nearly) room temperature. *manuscript*, 2019.
- 15 Lior Eldar and Aram W Harrow. Local hamiltonians whose ground states are hard to approximate. In *Foundations of Computer Science (FOCS), 2017 IEEE 58th Annual Symposium on*, pages 427–438. IEEE, 2017.
- 16 Shai Evra, Tali Kaufman, and Gilles Zémor. Decodable quantum ldpc codes beyond the  $\sqrt{n}$  distance barrier using high dimensional expanders. *arXiv preprint*, 2020. [arXiv:2004.07935](#).
- 17 Omar Fawzi, Antoine Grospellier, and Anthony Leverrier. Efficient decoding of random errors for quantum expander codes. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 521–534. ACM, 2018.
- 18 Michael H Freedman, David A Meyer, and Feng Luo. Z2-systolic freedom and quantum codes. *Mathematics of quantum computation, Chapman & Hall/CRC*, pages 287–320, 2002.
- 19 Oded Goldreich. Short locally testable codes and proofs: A survey in two parts. In *Property testing*, pages 65–104. Springer, 2010.
- 20 Matthew B Hastings. Trivial low energy states for commuting Hamiltonians, and the quantum PCP conjecture. *Quantum Information & Computation*, 13(5-6):393–429, 2013.
- 21 Matthew B Hastings. Quantum codes from high-dimensional manifolds. In *8th Innovations in Theoretical Computer Science Conference (ITCS 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.
- 22 Matthew B Hastings. Quantum codes from high-dimensional manifolds. In *LIPICs-Leibniz International Proceedings in Informatics*, volume 67. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.
- 23 Tali Kaufman, David Kazhdan, and Alexander Lubotzky. Isoperimetric inequalities for ramanujan complexes and topological expanders. *Geometric and Functional Analysis*, 26(1):250–287, 2016.
- 24 Tali Kaufman and Ran J Tessler. Quantum LDPC codes with  $\Omega(\sqrt{n} \log^k n)$  distance, for any  $k$ . *arXiv preprint*, 2020. [arXiv:2008.09495](#).
- 25 A Yu Kitaev. Fault-tolerant quantum computation by anyons. *Annals of Physics*, 303(1):2–30, 2003.
- 26 Anthony Leverrier, Vivien Londe, and Gilles Zémor. Towards local testability for quantum coding. *arXiv preprint*, 2019. [arXiv:1911.03069](#).

- 27 Anthony Leverrier, Jean-Pierre Tillich, and Gilles Zémor. Quantum expander codes. In *Foundations of Computer Science (FOCS), 2015 IEEE 56th Annual Symposium on*, pages 810–824. IEEE, 2015.
- 28 Anand Natarajan and Thomas Vidick. Low-degree testing for quantum states, and a quantum entangled games PCP for QMA. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 731–742. IEEE, 2018.
- 29 Chinmay Nirkhe, Umesh Vazirani, and Henry Yuen. Approximate low-weight check codes and circuit lower bounds for noisy ground states. In *45th International Colloquium on Automata, Languages, and Programming (ICALP 2018)*, volume 107, page 91. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2018.
- 30 David Poulin. Stabilizer formalism for operator quantum error correction. *Physical Review Letters*, 95(23):230504, 2005.





# Complete Problems for Multi-Pseudodeterministic Computations

Peter Dixon

Department of Computer Science, Iowa State University, Ames, IA, USA  
tooplark@iastate.edu

A. Pavan

Department of Computer Science, Iowa State University, Ames, IA, USA  
pavan@cs.iastate.edu

N. V. Vinodchandran

Department of Computer Science and Engineering, University of Nebraska, Lincoln, NE, USA  
vinod@cse.unl.edu

---

## Abstract

We exhibit several computational problems that are *complete* for multi-pseudodeterministic computations in the following sense: (1) these problems admit 2-pseudodeterministic algorithms (2) if there exists a pseudodeterministic algorithm for any of these problems, then any multi-valued function that admits a  $k$ -pseudodeterministic algorithm for a constant  $k$ , also admits a pseudodeterministic algorithm. We also show that these computational problems are complete for *Search-BPP*: a pseudodeterministic algorithm for any of these problems implies a pseudodeterministic algorithm for all problems in Search-BPP.

**2012 ACM Subject Classification** Theory of computation → Probabilistic computation; Theory of computation → Problems, reductions and completeness

**Keywords and phrases** Pseudodeterminism, Completeness, Collision Probability, Circuit Acceptance, Entropy Approximation

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.66

**Funding** Supported in part by NSF grants 1934884, 1849053, 1849048.

**Acknowledgements** We thank Oded Goldreich for comments and suggestions on an earlier draft of this paper. We also thank anonymous reviewers for helpful comments.

## 1 Introduction

Consider the search problem of producing a witness that two multi-variate polynomials  $f$  and  $g$  over a field are different. A simple probabilistic polynomial-time algorithm for this problem randomly picks an element  $t$  from the domain and outputs it if  $f(t) \neq g(t)$ . Even though this algorithm is simple and efficient and the error probability can be made arbitrarily small, Gat and Goldwasser [3] pointed out a deficiency: two different runs of the algorithm can produce two different witnesses with very high probability. Well-known probabilistic algorithms for several search problems, such as finding a large prime number or computing generators of cyclic groups, also exhibit this deficiency. This raises the question of whether we can design a probabilistic algorithm for search problems that will output the same witness on multiple executions, with high probability. Motivated by the above question, Gat and Goldwasser [3] introduced the notion of *pseudodeterministic algorithms*<sup>1</sup>. Informally, a probabilistic algorithm  $M$  is pseudodeterministic if for every  $x$ , there exists a

---

<sup>1</sup> Originally termed Bellagio algorithms



unique value  $v$  such that  $\Pr[M(x) = v]$  is high. Pseudodeterministic algorithms are appealing in several contexts, such as distributed computing and cryptography, where it is desirable that different invocations of an algorithm by different parties should produce the same output. In complexity theory, the notion of pseudodeterminism clarifies the relationship between search and decision problems in the context of randomized computation. It is not known whether derandomizing BPP to P implies derandomization of probabilistic search algorithms. However,  $\text{BPP} = \text{P}$  implies that pseudodeterministic search algorithms can be made deterministic [5].

## Prior Work

Since its introduction, pseudodeterminism and its generalizations have received considerable attention. In particular, designing pseudodeterministic algorithms for problems that admit polynomial-time randomized algorithms but without known deterministic algorithms continues to be a key line of research with some success. Gat and Goldwasser showed that there exist polynomial-time pseudodeterministic algorithms for algebraic problems such as finding quadratic non-residues and finding certificates that two multivariate polynomials are different [3]. Goldwasser and Grossman exhibited a pseudodeterministic NC algorithm for computing matchings in bipartite graphs [8]. Grossman designed a pseudodeterministic algorithm for computing primitive roots whose runtime matches the best known Las Vegas algorithm [11]. Oliveira and Santhanam [13] showed that there is a sub-exponential time pseudodeterministic algorithm for generating primes that works at infinitely many input lengths.

Subsequent works extended pseudodeterminism to several other scenarios. Works of Goemans, Goldwasser, Grossman, and Holden investigated pseudodeterminism in the context of interactive proofs [9, 4]. Goldwasser, Grossman, Mohanty and Woodruff [10] investigated pseudodeterminism in the data stream model. They showed that certain streaming problems admit faster pseudodeterministic algorithms in comparison to their deterministic counterparts. They also obtain space lower bounds for sketch based pseudodeterministic estimation of  $\ell_2$  norm. Goldreich, Goldwasser, and Ron [5] investigated pseudodeterminism in the context of sublinear-time algorithms. Dixon, Pavan, and Vinodchandran [2] studied pseudodeterminism in the context of approximation algorithms and showed that making Stockmeyer's [16] well-known approximate counting algorithm pseudodeterministic will yield new circuit lower bounds. Oliveira and Santhanam studied pseudodeterminism in the context of learning algorithms and showed that some randomized learning algorithms can be made pseudodeterministic under certain complexity theoretic assumptions [14]. Since then a few generalizations of pseudodeterminism such as *reproducible algorithms*, *influential bit algorithms* and *multi-pseudodeterministic algorithms* have been introduced [12, 7].

## Multi-Pseudodeterminism

Our main focus is on the notion of *multi-pseudodeterminism* recently introduced by Goldreich [7]. Consider the problem of estimating the average value of a function that is defined over a large but finite universe. It is well known that there is an efficient additive error, probabilistic approximation for this problem. So far, we do not know how to make this algorithm pseudodeterministic. Suppose that we relax the restriction of pseudodeterminism - instead of requiring that the algorithm outputs an unique approximation, the algorithm must output *one of two approximate values* with high probability. Then it is very easy to obtain such probabilistic algorithms [7]. Motivated by this, Goldreich

introduced the notion of *multi-pseudodeterminism*. For a positive integer  $k$ , a probabilistic algorithm  $A$  is  $k$ -pseudodeterministic if, for every input  $x$ , there exists a set  $S_x$  of size at most  $k$  such that  $A(x)$  belongs to  $S_x$  with probability at least  $\frac{k+1}{k+2}$ . Thus a pseudodeterministic algorithm is 1-pseudodeterministic. Goldreich's work established several key properties of multi-pseudodeterministic algorithms. This work showed that, as with the case of probabilistic and pseudodeterministic algorithms, error reduction is possible for multi-pseudodeterministic algorithms. Goldreich's work also established an equivalence between multi-pseudodeterminism and reproducible algorithms introduced by Grossman and Liu [12] and presented a composition result for multi-pseudodeterministic algorithms.

## Our Contributions

Our main focus is on the notion of *multi-pseudodeterminism*. The notion of multi-pseudodeterminism is especially interesting because there are computational problems that admit 2-pseudodeterministic algorithms for which we do not know of any pseudodeterministic algorithms. As mentioned earlier, it can be shown that randomized approximation algorithms can be made 2-pseudodeterministic (see Section 2). Thus it is significant to investigate the possibility of designing pseudodeterministic algorithms for problems that admit  $k$ -pseudodeterministic algorithms for small values of  $k$ .

Our main contribution is to show the existence of complete problems for multi-pseudodeterministic computations in the following sense: (1) these computational problems admit 2-pseudodeterministic algorithms, and (2) if there exists a pseudodeterministic algorithm for any of these problems, then all multi-valued functions that admit  $k$ -pseudodeterministic algorithms for a constant  $k$ , also admit pseudodeterministic algorithms.

The computational problems we consider are the following. We note that all of these problems admit 2-pseudodeterministic algorithms.

### ► Definition 1 (Computational Problems).

- COLLISION PROBABILITY ESTIMATION PROBLEM. *Given a Boolean circuit  $C : \{0, 1\}^n \rightarrow \{0, 1\}$ , give a  $(\varepsilon, \delta)$ -additive approximation of the collision probability of  $C$ .*
- ACCEPTANCE PROBABILITY ESTIMATION PROBLEM: *Given a Boolean circuit  $C : \{0, 1\}^n \rightarrow \{0, 1\}$ , give a  $(\varepsilon, \delta)$ -additive approximation for  $\Pr_{x \in U_n}[C(x) = 1]$ .*
- ENTROPY ESTIMATION PROBLEM: *Given a Boolean circuit  $C : \{0, 1\}^n \rightarrow \{0, 1\}$ , give an  $(\varepsilon, \delta)$ -additive approximation of the entropy of the distribution  $C(U_n)$ .*

We first show that if any of the above problems have a pseudodeterministic algorithm, then all 2-pseudodeterministic algorithms can be made pseudodeterministic and thus any  $(\varepsilon, \delta)$ -approximation algorithm for a function  $f$  can be made pseudodeterministic.

**Result 1:** *If any of the problems from Definition 1 admits a pseudodeterministic algorithm, every function that has a  $(\varepsilon, \delta)$ -approximation algorithm has a pseudodeterministic  $(\varepsilon, \delta)$ -approximation algorithm.*

Next we extend this result to  $k$ -pseudodeterminism.

**Result 2:** *If any of the problems from Definition 1 admits a pseudodeterministic algorithm, then any multivalued function that admits a  $k$ -pseudodeterministic algorithm also admits a pseudodeterministic algorithm.*

Note that the above result holds for any multivalued function computation. *Search problems* are multivalued functions whose outputs have an efficient verification procedure. Much of the work on pseudodeterminism focuses on problem in the class *Search-BPP*: search problems that have randomized algorithms whose outputs can be verified in BPP [6]. We extend the above result to problems in Search-BPP.

**Result 3:** *If any of the problems from Definition 1 admits a pseudodeterministic algorithm, then any problem in Search-BPP has a pseudodeterministic algorithm.*

Extending this result on Search-BPP to other classes of multivalued functions is an interesting question. Note that there are natural multivalued functions that admit efficient probabilistic algorithms but are not known to be in Search-BPP (because of the lack of an efficient verification procedure). For example the problem of outputting a Boolean function with high circuit complexity has a simple probabilistic algorithm but not known to be in Search-BPP.

## 2 Pseudodeterminism in Approximation Algorithms

In this section, we establish that COLLISION PROBABILITY ESTIMATION PROBLEM is complete. We first define the notions of pseudodeterminism and multi-pseudodeterminism for approximation algorithms formally. In general, an approximation algorithm can output different good approximations on different random choices. For an approximation algorithm  $A$  to be  $k$ -pseudodeterministic,  $A$  has to output with high probability at most  $k$  good approximations for any input.

► **Definition 2** (Multiplicative Approximation). *Let  $f$  be a function whose range is the integers. We say that a probabilistic algorithm  $A$  is an  $(\varepsilon, \delta)$ -multiplicative approximation algorithm for  $f$  if for every  $x$ , the random variable  $A(x)$  has the following property:  $\Pr \left[ \frac{f(x)}{(1+\varepsilon)} \leq A(x) \leq (1+\varepsilon)f(x) \right] \geq (1-\delta)$ . We say that  $A$  is an  $(\varepsilon, \delta)$   $k$ -pseudodeterministic multiplicative approximation algorithm for  $f$  if for every  $x$  there exists a set of integers  $V_x$  such that  $|V_x| \leq k$  and for every  $v \in V_x$*

$$\frac{f(x)}{(1+\varepsilon)} \leq v \leq (1+\varepsilon)f(x) \text{ and } \Pr[A(x) \in V_x] \geq 1 - \delta.$$

When  $k = 1$ , we call the algorithm pseudodeterministic.

► **Definition 3** (Additive Approximation). *Let  $f$  be a function whose range is  $[0, 1]$ . We say that a probabilistic polynomial-time algorithm  $A$  is an  $(\varepsilon, \delta)$ -additive approximation algorithm for  $f$  if for every  $x$ , the random variable  $A(x)$  has the following property:  $\Pr[(f(x) - \varepsilon) \leq A(x) \leq f(x) + \varepsilon] \geq (1 - \delta)$ . We say that  $A$  is an  $(\varepsilon, \delta)$   $k$ -pseudodeterministic additive approximation algorithm for  $f$  if for every  $x$  there exists a set  $V_x \subseteq [0, 1]$  such that for all  $v \in V_x$*

$$f(x) - \varepsilon \leq v \leq f(x) + \varepsilon \text{ and } \Pr[A(x) \in V_x] \geq 1 - \delta.$$

When  $k = 1$ , we call the algorithm pseudodeterministic.

► **Remark.** In general, for  $(\varepsilon, \delta)$  approximation algorithms (additive or multiplicative), error reduction is possible when  $\delta < 1/2$  and is bounded away from  $1/2$ . This is done by repeating the algorithm multiple times and taking the median value. Thus without loss of generality, we may assume that for every  $(\varepsilon, \delta)$  approximation algorithm,  $\delta \leq 1/2^n$ . Similarly, the error probability of  $(\varepsilon, \delta)$  pseudodeterministic approximation algorithms can be reduced to less than  $1/2^n$ .

Goldreich [7] observed that every additive error approximation algorithm can be made 2-pseudodeterministic; this extends to multiplicative approximation algorithms as well.

► **Proposition 4.** *Every optimization problem that admits an  $(\varepsilon, \delta)$  multiplicative approximation algorithm admits a  $(2\varepsilon + \varepsilon^2, \delta)$  2-pseudodeterministic multiplicative approximation algorithm. Similarly, every optimization problem that admits an  $(\varepsilon, \delta)$  additive approximation algorithm admits a  $(2\varepsilon, \delta)$  2-pseudodeterministic additive approximation algorithm.*

**Proof.** We give a proof for the multiplicative case. A similar argument holds for additive approximation algorithms. Let  $f$  be a function and let  $A$  be an  $(\varepsilon, \delta)$ -multiplicative approximation algorithm for  $f$ . The proposed 2-pseudodeterministic algorithm  $B$  will first run  $A(x)$  to get an approximation  $v$ .  $B$  outputs  $v_i$  from a set of points  $P$  (defined next) by choosing the smallest  $v_i \in P$  that is larger than  $v$ .  $P$  is defined as the set of points  $\{v_i \mid i \in \mathbb{N}\}$  where  $v_i = \lceil (1 + \varepsilon)^i \rceil$ .

Observe that  $v_{i+1} = (1 + \varepsilon)v_i$  for all  $i$ . For any input  $x$ ,  $A(x)$  outputs a value in the range  $\left[\frac{f(x)}{(1+\varepsilon)}, (1+\varepsilon)f(x)\right]$  and this range will contain at most 2 values from  $P$ . Hence  $B$  is 2-pseudodeterministic. Rounding up to the nearest value in  $P$  makes the approximation factor at most  $(1 + \varepsilon)^2 = 1 + 2\varepsilon + \varepsilon^2$ . ◀

## 2.1 Completeness of Collision Probability Estimation

In this section, we prove that COLLISION PROBABILITY ESTIMATION PROBLEM is complete for approximation algorithms in the context of pseudodeterminism. We start with the following observation.

► **Proposition 5.** COLLISION PROBABILITY ESTIMATION PROBLEM admits  $(\varepsilon, \delta)$  2-pseudodeterministic additive approximation algorithms for every  $\varepsilon < 1$  and  $\delta < 1$ .

**Proof.** We can estimate collision probability of  $C$  by generating  $O(1/\varepsilon^2 \log 1/\delta)$  independent pairs of strings  $\langle x_i, y_i \rangle$  and counting the number of times  $C(x_i) = C(y_i)$ . Simple application of Chernoff bound implies that this is a  $(\varepsilon, \delta)$  additive approximation algorithm. By the previous proposition, we can convert this algorithm into a 2-pseudodeterministic algorithm. ◀

We now prove the main theorem of the section.

► **Theorem 6.** *There exists  $\varepsilon' > 0$  such that if COLLISION PROBABILITY ESTIMATION PROBLEM has an  $(\varepsilon', \delta)$ -pseudodeterministic additive approximation algorithm, then every function  $f$  that admits a  $(\varepsilon, \delta)$  multiplicative approximation algorithm (resp. additive), has a  $(2\varepsilon + \varepsilon^2, \delta)$ -pseudodeterministic multiplicative approximation algorithm (resp. additive).*

**Proof.** We first provide intuition behind the proof. By Proposition 4, we can assume that  $f$  has a  $(\varepsilon', \delta)$  2-pseudodeterministic algorithm  $A$ , where  $\varepsilon' = 2\varepsilon + \varepsilon^2$ . The idea is to combine two strategies. For an input  $x$ , let  $a$  and  $b$  be the two good outputs of  $A(x)$ . Consider the case when one of the good outputs, say  $a$ , has a noticeably higher probability of occurrence than  $b$ . In that case we can run  $A$  several times and output the most frequent output. With high probability  $A(x)$  will output  $a$ . Another case is when both  $a$  and  $b$  appear with roughly equal probability. In this case, we can run  $A$  several times and output the smallest value. Since  $a$  and  $b$  appear with equal probability, the probability that in several runs of  $A$  we will see both  $a$  and  $b$  is very high and hence, this strategy will output  $\min\{a, b\}$ . The challenge is to decide which of these cases holds (note that these cases are not disjoint). However, if we have a pseudodeterministic algorithm for COLLISION PROBABILITY ESTIMATION PROBLEM, then we show that we can pseudodeterministically choose a good strategy. Now we provide details.

## 66:6 Complete Problems for Multi-Pseudodeterministic Computations

Let  $B$  be a pseudodeterministic algorithm that estimates collision probabilities of Boolean circuits. In particular, assume that  $B$  can pseudodeterministically estimate the collision probability of a circuit,  $\text{CP}(C)$ , with additive error  $1/49$  with probability  $\geq \frac{11}{12}$ . Without loss of generality we assume that the success probability of  $A$  is  $\geq 48/49$ . Assume that  $A$  uses  $m$  random bits on input  $x$ . Consider the following algorithm  $A'$ .

**Algorithm  $A'$ :** On input  $x$ ,  $A'(x)$  constructs a Boolean circuit  $C_x : \{0, 1\}^m \rightarrow \{0, 1\}$  that on input  $r$ , simulates  $A(x, r)$ , where  $A'(x)$  runs  $B$  on  $C_x$  to obtain an estimate of  $\text{CP}(C_x)$ . If  $B(C_x) \geq \frac{51}{98}$ ,  $A'$  runs  $A(x)$  *three* times and outputs the most frequent result. Otherwise, if  $B(C_x) < \frac{51}{98}$ ,  $A'$  runs  $A(x)$  16 times and outputs the smallest value.  $\lrcorner$

Now we will show that  $A'$  is a pseudodeterministic algorithm for  $f$ . Since  $A$  is 2-pseudodeterministic, there exists a set  $S_x$  of size at most 2 such that  $\Pr[A(x) \in S_x] \geq 48/49$  and every element in  $S_x$  is a  $(1 + \varepsilon')$  multiplicative approximation of  $f(x)$ . We first consider the case where the size of  $S_x$  is 1, say  $S_x = \{a\}$ . Note that in this case  $\text{CP}(C_x)$  is at least  $(48/49)^2 > 0.9$ . Thus  $A'(x)$  runs  $A$  three times and outputs the most frequent result, which is  $a$  with probability at least  $(48/49)^3 \geq 0.9$ . Thus  $A'(x)$  is pseudodeterministic. Thus, in the rest of the proof, assume that  $S_x = \{a, b\}$ . Let  $p = \Pr[A(x) = a]$  and  $q = \Pr[A(x) = b]$ . Assume without loss of generality that  $p \geq q$ . We first establish a relationship among  $p$ ,  $q$  and  $\text{CP}(C_x)$ .

▷ **Claim 7.** If  $\text{CP}(C_x) > \frac{25}{49}$ , then  $p > q + 1/7$

*Proof.* We prove the contrapositive: if  $p \leq q + 1/7$ , then  $\text{CP}(C_x) \leq \frac{25}{49}$ . Notice that  $\text{CP}(C_x)$  is maximized when  $p = q + \frac{1}{7}$  and  $\delta = 0$ , where  $\delta$  is the error probability of  $A$ . Since  $p + q + \delta = 1$ , it follows that  $p = 4/7$  and  $q = 3/7$ . Thus,  $\text{CP}(C_x) \leq (4/7)^2 + (3/7)^2 = 25/49$ .  $\triangleleft$

▷ **Claim 8.** If  $\text{CP}(C_x) < \frac{26}{49}$ , then  $q > \frac{1}{7}$

*Proof.* We prove the contrapositive: if  $q \leq \frac{1}{7}$ , then  $\Pr[\text{CP}(C_x)] \geq \frac{26}{49}$ . Note that  $\text{CP}(C_x)$  is minimized when  $p$  is as close to  $q$  as possible, and all other outputs are different from  $a$  and  $b$ . Then  $q = \frac{1}{7}$  and  $p = 1 - \frac{1}{7} - \frac{1}{49} = \frac{41}{49} \geq \frac{5}{7}$ . Thus  $\text{CP}(C_x) \geq (\frac{1}{7})^2 + (\frac{5}{7})^2 = \frac{26}{49}$ .  $\triangleleft$

▷ **Claim 9.** If  $p > q + 1/7$ , then the probability that in 3 independent runs  $A(x)$  outputs  $a$  at least twice is  $\geq 57/100$ .

*Proof.* The worst case is when  $p = q + 1/7$  and  $\delta = 1/49$ . In this case,  $p = \frac{55}{98}$  and  $q = \frac{41}{98}$ . The probability that  $A(x)$  outputs  $a$  at least twice in three runs is  $\geq \left(\frac{55}{98}\right)^3 + 3 \left(\frac{55}{98}\right)^2 \frac{44}{98} \geq \frac{57}{100}$ .  $\triangleleft$

▷ **Claim 10.** Let  $E$  be following event: “Among 16 independent runs of  $A(x)$ , every run outputs either  $a$  or  $b$  and at least one run outputs  $a$  and at least one run outputs  $b$ ”. If  $q \geq \frac{1}{7}$ , then  $\Pr[E] \geq 3/5$ .

*Proof.* Note that the probability of  $\overline{E}$  is at most the sum of the probabilities of i)  $A(x) \notin \{a, b\}$ , ii) Every run of  $A(x)$  outputs only  $a$ , and iii) Every run of  $A(x)$  outputs only  $b$ . This sum is maximized when  $q = 1/7$ ,  $\delta = 1/49$ ,  $p = 1 - 1/7 - 1/49$ . In this case,  $\Pr[\overline{E}] \leq 1 - (1 - \delta)^{16} p^{16} + q^{16} \leq \left(\frac{6}{7}\right)^{16} + \left(\frac{1}{7}\right)^{16} + 1 - \left(\frac{48}{49}\right)^{16} < \frac{2}{5}$ .  $\triangleleft$

▷ **Claim 11.**  $A'$  is pseudodeterministic.



Let  $y$  be the pseudodeterministic output of the probability estimator  $B$  on input  $C_x$ . If  $y > \frac{51}{98}$ , then  $\text{CP}(C_x) \geq \frac{25}{49}$ . By Claim 7,  $p > q + 1/7$ . By Claim 9, the probability that  $A(x)$  outputs  $a$  more often than  $b$  among three runs is at least  $\frac{57}{100}$ . Since  $y > \frac{51}{98}$ ,  $A'$  will run  $A$  3 times and output the most frequent result; this will output  $a$  with probability  $\geq \frac{57}{100}$ .

On the other hand, if  $y \leq \frac{51}{98}$ , then  $\text{CP}(C_x) \leq \frac{53}{98}$ . By Claim 8,  $q \geq 1/7$ . Since  $y \leq \frac{51}{98}$ ,  $A'$  will run  $A$  16 times and output the lexicographically smallest result. By Claim 10,  $\Pr[A \text{ outputs only } a \text{ and } b, \text{ and outputs } a \text{ and } b \text{ at least once}] \geq \frac{3}{5}$ , so  $\Pr[A' \text{ outputs } \min(a, b)] \geq \frac{3}{5}$ .

So, given that  $B$  outputs  $y$ ,  $A'$  will output one particular value  $x$  with probability  $\geq \frac{57}{100}$ .  $\Pr[A' \text{ outputs } x] \geq \Pr[B \text{ outputs } y] \Pr[A' \text{ outputs } x | B \text{ outputs } y] \geq \frac{11}{12} \frac{57}{100} \geq \frac{52}{100}$ . This can be increased to  $1 - \frac{1}{n}$  using standard amplification techniques. ◀

### 3 Pseudodeterminism for Multi-valued Functions

In this section, we generalize the results from the previous section to  $k$ -pseudodeterminism. Goldreich [7] defined the notion of  $k$ -pseudodeterminism for search problems and this definition can be extended to multivalued functions. A function  $f$  is multivalued if  $f(x)$  is a subset of the range (possibly empty set). Note that search problems can be cast as multivalued functions: Let  $R$  be a binary relation associated with a search problem, and define  $f(x)$  as the set of all  $y$  such that  $\langle x, y \rangle \in R$ .

► **Definition 12.** Let  $f$  be a multivalued function, i.e.,  $f(x)$  is a set. We say that  $f$  admits pseudodeterministic algorithms if there is a probabilistic polynomial-time algorithm  $A$  such that for every  $x$ , there exists a  $v \in f(x)$  such that  $A(x) = v$  with probability at least  $2/3$ . The function  $f$  admits  $k$ -pseudodeterministic algorithms if there is a probabilistic polynomial-time algorithm  $A$  such that for every  $x$ , there exists a set  $S_x \subseteq f(x)$  of size at most  $k$  and the probability that  $A(x) \in S(x)$  is at least  $\frac{k+1}{k+2}$ .

Goldreich [7] showed that if we threshold success probability to at least  $\frac{k+1}{k+2}$ , the success probability for  $k$ -pseudodeterministic algorithms can be amplified to  $1 - 1/2^{p(n)}$  for any polynomial  $p(\cdot)$ .

We show that if COLLISION PROBABILITY ESTIMATION PROBLEM can be made pseudodeterministic, then any  $k$ -pseudodeterministic algorithm for a multi-valued function problem can be made pseudodeterministic for a constant  $k$ . We first show how to reduce the size of the output set from  $k$  to  $k - 1$ .

► **Theorem 13.** If COLLISION PROBABILITY ESTIMATION PROBLEM has a  $(\varepsilon, \delta)$ -pseudodeterministic additive approximation algorithm with  $\varepsilon = 1/100$ , then for every multi-valued function  $f$  that admits a  $k$ -pseudodeterministic algorithm,  $f$  has a  $(k - 1)$ -pseudodeterministic algorithm.

**Proof.** Let  $B$  be a pseudodeterministic algorithm for COLLISION PROBABILITY ESTIMATION PROBLEM. In particular, assume that  $B$ , given a circuit  $C$ , estimates  $\text{CP}(C)$  to within  $\frac{1}{100}$  additive error with probability  $1 - \delta$ , where  $n$  is the length of the input to  $C$ .

Let  $A$  be a  $k$ -pseudodeterministic algorithm for a multi-valued function  $f$  with error probability  $\delta \leq \frac{1}{72k}$ . That is,  $A$ , on input  $x$ , outputs from a set  $S_x \subseteq f(x)$  of size  $\leq k$ , with probability  $\geq 1 - \frac{1}{72k}$ . We call the elements of  $S_x$  *good outputs* of  $A$ . Let  $m$  be the number of random bits used by  $A$ . We will design a  $(k - 1)$ -pseudodeterministic algorithm  $A'$  as follows:

**Algorithm  $A'$ :** On input  $x$ , first construct a circuit  $C_x$  that gets as input  $r$  and outputs  $A(x, r)$ . Then compute  $B(C_x)$ .

If  $B(C_x) < \frac{65}{100}$ , run  $A$   $n$  times on independent random bits. Output the *lexicographically smallest* element that appears at least  $\frac{n}{12k}$  times.

If  $B(C_x) \geq \frac{65}{100}$ , run  $A$   $n$  times using independent random bits. Output the most frequent value.  $\lrcorner$

As in the proof of Theorem 6, it is easy to see that if the size of  $S_x$  is 1,  $A'(x)$  is pseudodeterministic. Thus in the rest of the proof, we assume that the size of  $S_x$  is at least 2. For the proof of correctness, we first establish certain claims relating the collision probability of  $C_x$  and the behavior of  $A$ .

▷ **Claim 14.** If  $\text{CP}(C_x) \leq \frac{2}{3}$ , then there exist two *good* outputs  $a$  and  $b$  of  $A$  so that  $\Pr[A(x) = a] \geq \frac{1}{6k}$  and  $\Pr[A(x) = b] \geq \frac{1}{6k}$ .

*Proof.* Suppose only one good output  $a$  of  $A$  has probability  $\geq \frac{1}{6k}$ . Then that output has probability  $\geq \frac{5}{6} - \delta$ . This is because if the other  $(k-1)$  good outputs have probability  $< \frac{1}{6k}$ , the total probability of outputs other than  $a$  is  $< 1/6 + \delta$ . Thus  $\Pr[A(x) = a] \geq \frac{5}{6} - \delta$ .

$$\begin{aligned} \text{CP}(C_x) &\geq \Pr[A \text{ outputs } a \text{ on both runs}] \\ &\geq \left(\frac{5}{6} - \delta\right)^2 \\ &\geq \frac{25}{36} - 2\delta \geq \frac{2}{3} \end{aligned}$$

The last inequality holds since  $\delta \leq \frac{1}{72k}$  and  $k \geq 2$ .  $\triangleleft$

▷ **Claim 15.** Let  $a, b$  be the most and least likely good outputs of  $A$  with probabilities  $p$  and  $q$  respectively. If  $\text{CP}(C_x) > \frac{64}{100}$ , then  $p > q + \frac{1}{8k}$ .

*Proof.* If  $p \leq q + \frac{1}{8k}$ , then  $p \leq \frac{9}{8k} - \frac{\delta}{k} \leq \frac{9}{8k}$ . Then  $\text{CP}(C_x) \leq k \cdot \frac{81}{64k^2} + \delta^2 = \frac{81}{64k} + \delta^2$ . For  $k \geq 2$ , and  $\delta \leq \frac{1}{72k}$ , this quantity is  $\leq \frac{64}{100}$ .  $\triangleleft$

Now we will prove correctness of  $A'$ . On input  $x$ , let  $y$  be the pseudodeterministic output of  $B(C_x)$ . We will consider two cases:  $y \leq \frac{65}{100}$  and  $y > \frac{65}{100}$ .

**Case:  $y \leq \frac{65}{100}$**

▷ **Claim 16.** If  $y \leq \frac{65}{100}$ , then there exists a set  $S'_x(x) \subset S_x$  of size at most  $k-1$  such that

$$\Pr[A'(x) \in S'_x] \geq 1 - 2e^{-\frac{n}{72k^2}} - \delta$$

*Proof.* Note that  $B(C_x)$  outputs  $y$  with probability at least  $1 - \delta$ . Thus, if  $y \leq \frac{65}{100}$ , then with probability  $\geq 1 - \delta$ ,  $A'$  will run  $A$   $n$  times and output the lexicographically smallest result that appears at least  $\frac{n}{12k}$  times. Since  $y \leq \frac{65}{100}$ ,  $\text{CP}(C_x) \leq \frac{66}{100} < \frac{2}{3}$ . Therefore by Claim 14, there are at least 2 elements  $a$  and  $b$  from  $S_x$  that  $A$  outputs with probability  $\geq \frac{1}{6k}$ . Let  $b$  be the lexicographically larger of the two. We set  $S'_x = S_x \setminus \{b\}$ . Clearly  $S'_x$  contains at most  $k-1$  elements. Thus the probability that  $A'(x)$  outputs an element outside of  $S'_x$  is at most the sum of the probabilities of the following events: i)  $A'$  outputs  $b$  ii)  $A'$  outputs an element that is not in  $S_x$ , iii)  $B(C_x)$  does not output  $y$ .

We first bound the probability that  $A'$  outputs  $b$ . For  $A'$  to output  $b$ , it must be the case that in the  $n$  runs of  $A$  the value  $b$  is output at least  $n/12k$  times and the value  $a$  is output at most  $n/12k$  times. Thus

$$\Pr[A'(x) = b] \leq \Pr[A(x) \text{ outputs } a \text{ less than } n/12k \text{ times among } n \text{ runs}]$$

Since the probability that  $A(x)$  outputs  $a$  is least  $n/6k$ , the expected number of times  $A(x)$  outputs  $a$  in  $n$  runs is  $\geq \frac{n}{6k}$ . Thus by Chernoff bound, this is at most  $e^{-n/72k^2}$ . We now bound the probability of the second event. The probability that  $A(x)$  does not belong to  $S_x$  is at most  $1/72k$ . For  $A'(x)$  to output an element  $c$  that is not in  $S_x$ , it must be the case that  $c$  is output  $\geq n/12k$  times among  $n$  runs of  $A$ . Again by Chernoff bound, this is at most  $e^{-n/18k^2}$ . Finally, the probability that  $B(C_x)$  does not output  $y$  is at most  $\delta$ . Thus  $A'(x) \in S'_x$  with probability at least  $1 - 2e^{-n/72k^2} - \delta$ .  $\triangleleft$

**Case:**  $y > \frac{65}{100}$

$\triangleright$  **Claim 17.** If  $y > \frac{65}{100}$ , there exists a sets  $S'_x \subseteq S_x$  of size at most  $k - 1$  such that

$$\Pr[A'(x) \in S'_x] \geq 1 - 2e^{-\frac{n}{8k}} - \delta$$

*Proof.* If  $y > \frac{65}{100}$ , then with probability  $\geq 1 - \delta$ ,  $A'$  will run  $A$   $n$  times and output the most frequent result. Also,  $\text{CP}(C_x) > \frac{64}{100}$ . By Claim 15,  $p > q + \frac{1}{8k}$  where  $p$  is the probability of the most likely element  $a$  from  $S_x$  and  $q$  is the probability of least likely element  $b$  from  $S_x$ . We define  $S'_x$  as  $S_x - \{b\}$ . As before, the probability that the output of  $A'$  does not belong to  $S'_x$  is at most the sum of the probabilities of: i)  $A'(x)$  outputs  $b$  ii) the output of  $A'(x)$  does not belong to  $S_x$  iii)  $B(C_x)$  does not output  $y$ .

We will first analyze the probability of the event that  $A'$  outputs  $b$ . For this, consider the event  $E = 'A \text{ outputs } b \text{ more often than } a \text{ in } n \text{ trials}'$ . Clearly, the probability that  $A'(x)$  outputs  $b$  is at most  $\Pr[E]$ . Define random variables  $X_i$  that take value 0 if  $A(x)$  outputs  $a$  in  $i^{\text{th}}$  run, 1 if  $A(x)$  outputs  $b$  in the  $i^{\text{th}}$  run, and  $1/2$  otherwise. Note that  $E[X_i] = \frac{1}{2} - \frac{p-q}{2}$ , which is at most  $\frac{1}{2} - \frac{1}{16k}$ . Let  $X = \sum_{i=1}^n X_i$ . Now,  $\Pr[E]$  is  $\Pr[X > n/2]$ .

$$\begin{aligned} \Pr[X > n/2] &= \Pr\left[\frac{\sum X_i}{n} > 1/2\right] \\ &\leq \Pr\left[\left|\frac{\sum X_i}{n} - E[X_i]\right| \geq \frac{1}{16k}\right] \\ &\leq e^{-n/256k^2} \text{ by Chernoff bound} \end{aligned}$$

To bound the probability of the second event, consider the probability that  $A(x)$  outputs an element not in  $S_x$  more frequently than  $a$  in  $n$  runs. Since the probability that  $A(x)$  outputs an element that is not in  $S_x$  is at most  $1/72k$ , by the same argument the probability of this event is at most  $e^{-n/256k^2}$ . Finally the probability that  $B(C_x)$  does not output  $y$  is at most  $\delta$ . The claim follows.  $\triangleleft$

Combining Claims 16 and 17, we have that  $A'$  outputs a value from  $S'_x$  with probability at least  $1 - 2e^{-n/72k^2} - \delta \geq \frac{k}{k+1}$  for large enough  $n$ , since  $\delta$  can be made exponentially small.  $\blacktriangleleft$

We now state the main result of this section.

► **Theorem 18.** *If COLLISION PROBABILITY ESTIMATION PROBLEM has a  $(\varepsilon, \delta)$ -pseudodeterministic additive approximation algorithm with  $\varepsilon = 1/100$ , every multi-valued function  $f$  that admits a  $k$ -pseudodeterministic algorithm, has a pseudodeterministic algorithm.*

**Proof.** Let  $B$  be the pseudodeterministic algorithm for COLLISION PROBABILITY ESTIMATION PROBLEM, where  $B$  runs in time  $n^c$  with error probability  $\leq \delta$ . To convert a  $k$ -pseudodeterministic algorithm to a pseudodeterministic algorithm, we repeatedly apply Theorem 13. We start with a  $k$ -pseudodeterministic algorithm  $A_k$  whose runtime is bounded by  $n^t$ . On input  $x$ ,  $A_{k-1}$  constructs  $C_{A_k, x}$  with size  $\leq 4n^{2t}$ .  $A_{k-1}$  computes  $B(C)$ , which takes  $\leq 4^c n^{2tc}$  time, then runs  $A_k(x)$   $n$  times. In total,  $A_{k-1}(x)$  takes  $\leq n^{t+1} + 4^c n^{2tc} \leq n^{4tc}$ . Applying this conversion  $(k-1)$  times, we obtain  $A_1$ , a pseudodeterministic algorithm with runtime of  $O(n^{t(4c)^k})$ . Since  $k$  is a constant the runtime is polynomial. Note that in each iteration, the error probability remains the same. Thus  $A_1$  is pseudodeterministic. ◀

### 3.1 Circuit Probability Acceptance

In this subsection we observe the equivalence of COLLISION PROBABILITY ESTIMATION PROBLEM and ACCEPTANCE PROBABILITY ESTIMATION PROBLEM in the context of pseudodeterminism.

► **Proposition 19.** *There exist  $\varepsilon, \varepsilon' > 0$  such that COLLISION PROBABILITY ESTIMATION PROBLEM has an  $(\varepsilon, \delta)$ -pseudodeterministic additive approximation algorithm if and only if ACCEPTANCE PROBABILITY ESTIMATION PROBLEM has an  $(\varepsilon', \delta)$ -pseudodeterministic additive approximation algorithm.*

**Proof.** It is easy to see that ACCEPTANCE PROBABILITY ESTIMATION PROBLEM admits an  $(\varepsilon, \delta)$  additive approximation algorithm. Thus by Proposition 4, it has a 2-pseudodeterministic  $(\varepsilon, \delta)$  approximation algorithm. By Theorem 6, if COLLISION PROBABILITY ESTIMATION PROBLEM admits a pseudodeterministic algorithm, then ACCEPTANCE PROBABILITY ESTIMATION PROBLEM admits a pseudodeterministic algorithm.

Let  $B$  be a pseudodeterministic algorithm for ACCEPTANCE PROBABILITY ESTIMATION PROBLEM. Consider the following algorithm to estimate the collision probability of a circuit  $C$ : If  $B(C)$  outputs  $v$ , output  $v^2 + (1-v)^2$ . Let  $p = \Pr[C(U_n) = 1]$ . If  $v \in (p - \varepsilon, p + \varepsilon)$ , then the output of  $B$  belongs to  $(\text{CP}(C) - 8\varepsilon, \text{CP}(C) + 8\varepsilon)$ . Clearly  $B$  is pseudodeterministic. ◀

The following result is a corollary of the above proposition and Theorem 18.

► **Theorem 20.** *There exists  $\varepsilon' > 0$  such that if ACCEPTANCE PROBABILITY ESTIMATION PROBLEM admits an  $(\varepsilon', \delta)$  pseudodeterministic additive approximation algorithm, then every function  $f$  that admits an  $(\varepsilon, \delta)$  multiplicative approximation algorithm has a  $(3\varepsilon, \delta)$ -pseudodeterministic multiplicative approximation algorithm.*

### 3.2 Entropy Estimation

In this subsection we show that ENTROPY ESTIMATION PROBLEM and ACCEPTANCE PROBABILITY ESTIMATION PROBLEM are equivalent in the context of pseudodeterminism. We first observe that ENTROPY ESTIMATION PROBLEM admits an  $(\varepsilon, \delta)$ , 2-pseudodeterministic additive approximation algorithm.

► **Proposition 21.** *There is an  $(\varepsilon, \delta)$  2-pseudodeterministic approximation algorithm for the ENTROPY ESTIMATION PROBLEM.*

**Proof.** Given a circuit  $C$ , let  $p = \Pr[C(U_n) = 1]$ . As in Proposition 5, compute an approximate value  $q$  of  $p$  and output  $H(q)$ . It follows that  $H(q)$  is an approximation of  $H(p)$  due to the known result that entropy can be approximated by the empirical distribution obtained from sampling, for example see [15, 1]. By Proposition 4, we obtain a 2-pseudodeterministic algorithm.  $\blacktriangleleft$

Note that the above proof yields the following.

► **Proposition 22.** *There exist  $\varepsilon$  and  $\varepsilon'$  such that if ACCEPTANCE PROBABILITY ESTIMATION PROBLEM has an  $(\varepsilon, \delta)$  pseudodeterministic additive error approximation algorithm, then ENTROPY ESTIMATION PROBLEM has  $(\varepsilon', \delta)$  pseudodeterministic, additive error approximation algorithm.*

Next we reduce ACCEPTANCE PROBABILITY ESTIMATION PROBLEM to ENTROPY ESTIMATION PROBLEM. Thus a pseudodeterministic algorithm for ENTROPY ESTIMATION PROBLEM implies a pseudodeterministic algorithm for ACCEPTANCE PROBABILITY ESTIMATION PROBLEM.

The main technical result that we show is that an approximation of the entropy of  $C(U_n)$  can be used to approximate the probability that  $C(U_n) = 1$ . It is possible that this technical result is known or is a folklore; we could not find a reference. Thus, for completeness a proof is provided in the appendix.

► **Theorem 23.** *Suppose that there is a  $(\varepsilon, \delta)$  pseudodeterministic approximation algorithm for ENTROPY ESTIMATION PROBLEM for a sufficiently small  $\varepsilon$ . Then there is a  $(1/100, \delta + e^{-O(n)})$  pseudodeterministic approximation algorithm for ACCEPTANCE PROBABILITY ESTIMATION PROBLEM.*

Using Proposition 19 and Theorem 23, we obtain that both ACCEPTANCE PROBABILITY ESTIMATION PROBLEM and ENTROPY ESTIMATION PROBLEM are complete for  $k$ -pseudodeterministic computations.

► **Theorem 24.** *There exist  $\varepsilon > 0$ , such that if either of ACCEPTANCE PROBABILITY ESTIMATION PROBLEM or ENTROPY ESTIMATION PROBLEM admit  $(\varepsilon, \delta)$ -pseudodeterministic, additive approximation algorithm, then every multivalued function that has a  $k$ -pseudodeterministic algorithm has a pseudodeterministic algorithm.*

## 4 Pseudodeterminism for Search Problems

In this section we show that if any of the 3 computational problems we consider has pseudodeterministic approximation schemes then every problem in Search-BPP has pseudodeterministic algorithms. The class Search-BPP was formally introduced by Goldreich [6]

► **Definition 25** (Search BPP [6]). *A search problem is a relation  $R \subseteq \{0, 1\}^* \times \{0, 1\}^*$ . For every  $x$ , the witness set  $W_x$  of  $x$  with respect to  $R$  is  $\{y \mid (x, y) \in R\}$ . A search problem  $R$  is in search-BPP (1) if there exists a probabilistic polynomial-time algorithm  $A$  such that for every  $x$  for which  $W_x \neq \emptyset$ ,  $A(x) \in W_x$  with probability  $\geq 2/3$ , (2) and there exists a probabilistic polynomial time algorithm  $B$  such that if  $(x, y) \in R$ , then  $B(x, y)$  accepts with probability  $> 2/3$ , and if  $(x, y) \notin R$  then  $B(x, y)$  accepts with probability  $< 1/3$ .*

We will first show that if ACCEPTANCE PROBABILITY ESTIMATION PROBLEM has a pseudodeterministic, additive, approximation scheme, then Search-BPP problems can be made pseudodeterministic. Then we will use Theorem 20 to prove that if ACCEPTANCE

## 66:12 Complete Problems for Multi-Pseudodeterministic Computations

PROBABILITY ESTIMATION PROBLEM has a  $(\epsilon, \delta)$  pseudodeterministic approximation algorithm, then Search-BPP problems admit pseudodeterministic algorithms. We first recall the definition of approximation scheme.

► **Definition 26.** A function  $f : \Sigma^* \rightarrow \mathbb{Q}$  has an additive, approximation scheme if there is a probabilistic polynomial time algorithm  $A$  that gets  $x, \epsilon$ , and  $\delta$  as input and

$$\Pr[f(x) - \epsilon \leq A(x, \epsilon, \delta) \leq f(x) + \epsilon] \geq 1 - \delta$$

► **Theorem 27.** If ACCEPTANCE PROBABILITY ESTIMATION PROBLEM has a pseudodeterministic, additive, approximation algorithm scheme, then every problem in Search-BPP has a pseudodeterministic algorithm.

**Proof.** Let  $R$  be a problem in Search-BPP and let  $A$  and  $B$  be probabilistic algorithms that witness  $R$  in search-BPP according to the definition.

The idea is to use the method of conditional probabilities to construct a good random choice  $z_A$  for  $A$  on input  $x$  first, and then output  $A(x, z_A)$ . The search for  $z_A$  will be aided by the pseudodeterministic approximation algorithm for ACCEPTANCE PROBABILITY ESTIMATION PROBLEM.

Consider the following probabilistic algorithm  $B'$  that, on input  $x$  of length  $n$ , first simulates  $A$  to get an output  $y$  and then runs  $B(x, y)$  and accepts if  $B$  accepts. Then  $\Pr[B'(x) \text{ accepts}] \geq 2/5$ . Let  $m = p(n)$  be the polynomial bounding the length of the random string of  $B'$ . We will view the random string  $r$  that  $B'$  uses as  $r_A r_B$  where  $r_A$  is the random string that  $B'$  uses to simulate  $A$  and  $r_B$  is to simulate  $B$ . Let  $A_{\text{ape}}$  be a pseudodeterministic approximation algorithm for ACCEPTANCE PROBABILITY ESTIMATION PROBLEM. We will use  $A_{\text{ape}}$  with error  $\epsilon \leq \frac{1}{n \cdot p(n)}$  and confidence  $1 - \delta \geq 1 - \frac{1}{n \cdot p(n)}$ .

For an input  $x$  let  $C(r_A r_B)$  be the Boolean circuit that simulates  $B'$  on  $x$  using random string  $r = r_A r_B$  and outputs 1 if and only if  $B'$  accepts  $x$  on  $r$ . Thus for any  $x$  where  $W_x \neq \emptyset$ ,  $\Pr[C = 1] \geq 2/5$ . For a binary string  $z \in \{0, 1\}^l$ , let  $C_z : \{0, 1\}^{m-l} \rightarrow \{0, 1\}$  be the circuit obtained by fixing the first  $l$  bits of  $C$ 's input to  $z$ . We now describe the algorithm  $A_R$  for the search problem that pseudodeterministically outputs a  $y \in W_x$ .

**Algorithm PseudoA<sub>R</sub>:** On input  $x$ , construct the circuit  $C$  that gets  $r = r_A r_B$  as input and outputs 1 if  $B'$  accepts  $(x, r)$  on random string  $r = r_A r_B$ . Initialize  $z = \lambda$ , the empty string. Iterate from  $i = 1$  to  $m = p(n)$ . At the  $i^{\text{th}}$  iteration, simulate  $A_{\text{ape}}(C_z 0)$  (pseudodeterministically) to approximate  $\Pr[C_{z0}(r) = 1]$  up to an additive error  $\epsilon$  and confidence  $(1 - \delta)$  to get a value  $v$ . If  $v \geq 2/5 - (2i + 1)\epsilon$  then  $z \leftarrow z0$  otherwise  $z \leftarrow z1$ . Continue to the next iteration. After the  $m^{\text{th}}$  iteration let  $z = z_A z_B$  be the binary string of length  $m$  constructed. Output  $A(x, z_A)$ .  $\lrcorner$

**Correctness:** Since error probability of  $A_{\text{ape}}$  is  $\leq \frac{1}{n \cdot m}$ , and we are making  $m$  calls to  $A_{\text{ape}}$ , by the union bound, the probability that any one of the calls makes an error is  $\leq 1/n$ . For the rest of the argument we assume all the calls to  $A_{\text{ape}}$  pseudodeterministically output an approximation to acceptance probability within an additive error of  $\epsilon$ .

► **Claim 28.** For every  $i$ , for the string  $z$  constructed at the end of the  $i^{\text{th}}$  iteration,  $\Pr[C_z = 1] \geq \frac{2}{5} - 2i\epsilon$ .

**Proof.** We prove this by induction on  $i$ . For  $i = 0$ , the hypothesis holds since  $\Pr[C = 1] \geq 2/5$ . Assume the hypothesis holds for  $i$ . Consider the  $(i + 1)^{\text{th}}$  iteration. Using conditional probabilities, after the  $i^{\text{th}}$  iteration,  $\Pr[C_{z0} = 1] \geq \frac{2}{5} - 2i\epsilon$  or  $\Pr[C_{z1} = 1] \geq \frac{2}{5} - 2i\epsilon$ . Suppose at the  $(i + 1)^{\text{th}}$  iteration the value  $v$  returned by  $A_{\text{ape}}(C_{z0})$  is  $\geq 2/5 - (2i + 1)\epsilon$ . Then  $z$



is updated to  $z_0$  by the algorithm and from the approximation guarantee of  $A_{ape}$  we have that  $\Pr[C_{z_0} = 1] \geq 2/5 - (2i + 1)\epsilon - \epsilon = 2/5 - 2(i + 1)\epsilon$ . On the other hand suppose  $v < 2/5 - (2i + 1)\epsilon$ . Then  $z$  is updated to  $z_1$ . Also  $\Pr[C_{z_0} = 1] < v + \epsilon = 2/5 - 2i\epsilon$  and hence  $\Pr[C_{z_1} = 1] \geq 2/5 - 2i\epsilon \geq 2/5 - 2(i + 1)\epsilon$   $\triangleleft$

Thus for any  $1 \leq i \leq m$   $\Pr[C_z = 1] \geq 2/3 - 1/n$  and hence the algorithm outputs a  $z$  so that  $B'(x, z)$  accepts. Hence the output of the algorithm  $A(x, z_A) \in W_x$ .

The algorithm  $\text{Pseudo}A_R$  can be seen as a deterministic algorithm making subroutine calls to the pseudodeterministic algorithm  $A_{ape}$ . Hence the overall algorithm is pseudodeterministic. The probability of error is bounded by any one of the calls to  $A_{ape}$  making an error which is  $\leq 1/n$ .  $\blacktriangleleft$

Next we will show that if ACCEPTANCE PROBABILITY ESTIMATION admits  $(\epsilon, \delta)$  pseudodeterministic additive approximation, then admits pseudodeterministic additive approximation scheme.

► **Proposition 29.** *There exists  $\epsilon > 0$  such that if ACCEPTANCE PROBABILITY ESTIMATION admits an  $(\epsilon, \delta)$  pseudodeterministic additive approximation, then it admits a pseudodeterministic approximation scheme.*

**Proof.** We first note that ACCEPTANCE PROBABILITY ESTIMATION PROBLEM admits an additive, approximation scheme. By Theorem 20, there is an  $\epsilon' > 0$  such that if ACCEPTANCE PROBABILITY ESTIMATION PROBLEM has an  $(\epsilon', \delta)$  pseudodeterministic approximation algorithm, then every  $(\epsilon, \delta)$ -additive approximation algorithm for a function  $f$  can be made into a  $(3\epsilon, \delta)$  pseudodeterministic, additive approximation algorithm. The same proof shows that if  $f$  admits an approximation scheme, then it can be made into a pseudodeterministic approximation scheme.  $\blacktriangleleft$

The main result of this section is a corollary of the above proposition and Theorem 27.

► **Theorem 30.** *There exists  $\epsilon > 0$  such that if ACCEPTANCE PROBABILITY ESTIMATION PROBLEM has a  $(\epsilon, \delta)$  pseudodeterministic approximation algorithm, then every problem in Search-BPP has a pseudodeterministic algorithm.*

## References

- 1 Jayadev Acharya, Sourbh Bhadane, Piotr Indyk, and Ziteng Sun. Estimating entropy of distributions in constant space. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 5163–5174, 2019.
- 2 Peter Dixon, A. Pavan, and N. V. Vinodchandran. On pseudodeterministic approximation algorithms. In *43rd International Symposium on Mathematical Foundations of Computer Science, MFCS 2018, August 27-31, 2018, Liverpool, UK*, volume 117 of *LIPIcs*, pages 61:1–61:11, 2018.
- 3 E. Gat and S. Goldwasser. Probabilistic search algorithms with unique answers and their cryptographic applications. *Electronic Colloquium on Computational Complexity (ECCC)*, 18:136, 2011.
- 4 Michel Goemans, Shafi Goldwasser, and Dhiraj Holden. Doubly-efficient pseudo-deterministic proofs. *arXiv*, 2019. [arXiv:1910.00994](https://arxiv.org/abs/1910.00994).
- 5 O. Goldreich, S. Goldwasser, and D. Ron. On the possibilities and limitations of pseudodeterministic algorithms. In *Innovations in Theoretical Computer Science, ITCS '13, Berkeley, CA, USA, January 9-12, 2013*, pages 127–138, 2013.
- 6 Oded Goldreich. In a world of P=BPP. In Oded Goldreich, editor, *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation*, volume 6650 of *Lecture Notes in Computer Science*, pages 191–232. Springer, 2011.



- 7 Oded Goldreich. Multi-pseudodeterministic algorithms. *Electronic Colloquium on Computational Complexity (ECCC)*, 26:12, 2019.
- 8 S. Goldwasser and O. Grossman. Bipartite perfect matching in pseudo-deterministic NC. In *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, pages 87:1–87:13, 2017.
- 9 S. Goldwasser, O. Grossman, and D. Holden. Pseudo-deterministic proofs. *CoRR*, abs/1706.04641, 2017.
- 10 Shafi Goldwasser, Ofer Grossman, Sidhanth Mohanty, and David P. Woodruff. Pseudo-deterministic streaming. In Thomas Vidick, editor, *11th Innovations in Theoretical Computer Science Conference, ITCS*, volume 151 of *LIPIcs*, pages 79:1–79:25, 2020.
- 11 O. Grossman. Finding primitive roots pseudo-deterministically. *Electronic Colloquium on Computational Complexity (ECCC)*, 22:207, 2015.
- 12 Ofer Grossman and Yang P. Liu. Reproducibility and pseudo-determinism in log-space. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 606–620. SIAM, 2019.
- 13 I. Oliveira and R. Santhanam. Pseudodeterministic constructions in subexponential time. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 665–677, 2017.
- 14 Igor Carboni Oliveira and Rahul Santhanam. Pseudo-derandomizing learning and approximation. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2018*, volume 116 of *LIPIcs*, pages 55:1–55:19, 2018.
- 15 Liam Paninski. Estimating entropy on  $m$  bins given fewer than  $m$  samples. *IEEE Trans. Inf. Theory*, 50(9):2200–2203, 2004.
- 16 L. Stockmeyer. The complexity of approximate counting (preliminary version). In *Proceedings of the 15th Annual ACM Symposium on Theory of Computing, 25-27 April, 1983, Boston, Massachusetts, USA*, pages 118–126, 1983.

## A Proof of Theorem 23

We first start with the following lemma. Here  $H$  is the binary entropy function  $H(p) = -p \log p - (1-p) \log(1-p)$ .

▷ **Claim 31.** Let  $0 \leq a + b \leq 1$ .  $H(a + b) \geq H(a) + b \log \frac{1-a-b}{a+b}$

Proof.

$$\begin{aligned}
 H(a + b) &= -(a + b) \log(a + b) - (1 - a - b) \log(1 - a - b) \\
 &= -a \log(a + b) - (1 - a) \log(1 - a - b) - b \log(a + b) - (-b) \log(1 - a - b) \\
 &\geq -a \log(a) - (1 - a) \log(1 - a) + b(\log(1 - a - b) - \log(a + b)) \\
 &= H(a) + b \log \frac{1 - a - b}{a + b}
 \end{aligned}$$

where the third line follows by Gibbs' inequality. ◁

We now provide a proof of Theorem 23.

**Proof of Theorem 23.** Suppose  $A$  is a pseudodeterministic algorithm that, given a Boolean circuit  $C$ , outputs  $(\varepsilon, \delta)$  approximation of  $H(C(U_n))$ . Let  $r = \Pr[C(U_n) = 1]$ . Our goal is to design a pseudodeterministic algorithm to estimate  $r$ . Let  $q$  be the smaller of  $1 - r$  and  $r$ . We will first design a pseudodeterministic algorithm  $B$  that outputs a value  $p$  such that  $p$  is within  $1/100$  of  $q$ .

$B(C)$  runs  $A(C)$  to obtain  $y$ . If  $y \geq H(\frac{1}{2} - \frac{1}{100}) + \varepsilon$ , then output  $\frac{1}{2}$ . Otherwise, do a binary search for  $p$  in the range  $(0, 1/2)$  such that  $-p \log p - (1-p) \log(1-p)$  lies within  $[y, y + 1/2^n)$ . We consider two cases.

**Case 1:**  $y \geq H(\frac{1}{2} - \frac{1}{100}) + \varepsilon$ . In this case, then clearly  $\frac{1}{2} - \frac{1}{100} \leq r \leq \frac{1}{2} + \frac{1}{100}$ .  $B(C)$  will output  $\frac{1}{2}$ , which is within  $\frac{1}{100}$  of  $r = \Pr[C = 1]$ , with probability  $\geq 1 - \delta$ .

**Case 2:**  $y < H(\frac{1}{2} - \frac{1}{100}) + \varepsilon$ . Since  $|H(p) - y| \leq 1/2^n$  and  $|H(q) - y| \leq \varepsilon$ , we have that  $|H(p) - H(q)| \leq \varepsilon + 1/2^n = \varepsilon'$ . Now we bound how far  $p$  is from  $q$ . For this we need the following two technical claims.

▷ **Claim 32.** Let  $a \leq \frac{1}{2}$ . If  $H(a) \leq H(\frac{1}{2} - \frac{1}{100}) + \varepsilon'$ , then  $a \leq \frac{1}{2} - \frac{1}{100} + \frac{1}{c}$ , for any  $c$  that satisfies  $\varepsilon' \leq \frac{1}{2c} \log(\frac{102c-200}{98c+200})$ .

Proof.

$$\begin{aligned} H(a) &\leq H\left(\frac{1}{2} - \frac{1}{100}\right) + 2\varepsilon \\ &\leq H\left(\frac{1}{2} - \frac{1}{100}\right) + \frac{1}{c} \log\left(\frac{102c-200}{98c+200}\right) \\ &= H\left(\frac{1}{2} - \frac{1}{100}\right) + \frac{1}{c} \log\left(\frac{\frac{1}{2} + \frac{1}{100} - \frac{1}{c}}{\frac{1}{2} - \frac{1}{100} + \frac{1}{c}}\right) \\ &\leq H\left(\frac{1}{2} - \frac{1}{100} + \frac{1}{c}\right) \text{ by Claim 31} \end{aligned}$$

Thus  $a \leq \frac{1}{2} - \frac{1}{100} + \frac{1}{c}$ . ◁

▷ **Claim 33.** Let  $a = b + \ell$  where  $a \leq \frac{1}{2}$ ,  $a, b, \ell \geq 0$ . If  $H(a) - H(b) \leq \varepsilon'$  and  $a \leq \frac{1}{2} - \frac{1}{100} + \frac{1}{c} \leq \frac{1}{2}$ , then  $\ell \leq \frac{\varepsilon'}{\log \frac{102c-200}{98c+200}}$ .

Proof.

$$\begin{aligned} \varepsilon' &\geq H(a) - H(b) \\ &= H(b + \ell) - H(b) \\ &\geq H(b) + \ell \log\left(\frac{1-b-\ell}{b+\ell}\right) - H(b) \text{ (By Claim 31)} \\ &= \ell \log\left(\frac{1-a}{a}\right) \end{aligned}$$

Since  $a \leq \frac{1}{2}$ , the minimum value for this is when  $a$  is as close to  $\frac{1}{2}$  as possible.

$$\begin{aligned} &\geq \ell \log \frac{1 - (\frac{1}{2} - \frac{1}{100} + \frac{1}{c})}{(\frac{1}{2} - \frac{1}{100} + \frac{1}{c})} \\ &= \ell \log \frac{102c-200}{98c+200} \\ \Rightarrow \ell &\leq \frac{\varepsilon'}{\log \frac{102c-200}{98c+200}} \end{aligned} \quad \text{◁}$$

▷ **Claim 34.** If  $\varepsilon'$  is sufficiently small, then there is a constant  $c$  satisfying

$$\frac{\varepsilon'}{\log \frac{102c-200}{98c+200}} \leq \frac{1}{100} \text{ and } \varepsilon' \leq \frac{1}{2c} \log\left(\frac{102c-200}{98c+200}\right)$$

## 66:16 Complete Problems for Multi-Pseudodeterministic Computations

Proof. It can be verified that when  $\varepsilon' = 1/42000$  and  $c = 1100$ , the above inequalities are satisfied.  $\triangleleft$

Now we are ready to prove that  $|p - q| \leq \frac{1}{100}$

▷ Claim 35.  $|p - q| \leq \frac{1}{100}$

Proof. First, suppose  $p > q$ . Then  $p = q + \ell$ . Since,  $y < H(\frac{1}{2} - \frac{1}{100}) + \varepsilon$  and  $|y - H(p)| \leq \frac{1}{2^n}$ , so  $H(p) \leq H(\frac{1}{2} - \frac{1}{100}) + \varepsilon + \frac{1}{2^n} = H(\frac{1}{2} - \frac{1}{100}) + \varepsilon'$ . By Claim 34, we have that  $c = 1/1100$ . By claim 32,  $p \leq \frac{1}{2} - \frac{1}{100} + \frac{1}{c}$ . By claim 33, we obtain that  $\ell \leq \frac{\varepsilon'}{\log \frac{102c-200}{98c+200}} \leq 1/100$ , thus  $p \leq q + 1/100$ . A similar argument shows that if  $p < q$ , then  $p \geq q - 1/100$ .  $\triangleleft$

We found a value  $p$  that is  $1/100$ -close to  $q$ , and the goal is to estimate  $r = \Pr[C(U_n) = 1]$ , where  $q = \min\{r, 1 - r\}$ . Thus  $p$  is either close to  $r$  or to  $1 - r$ . Now we run  $C(U_n)$ ,  $n$  times; if there are more 1s than 0s output  $1 - p$ ; else output  $p$ . Using Chernoff bounds, it follows that the output is  $1/100$ -close to  $q$  with probability  $\leq 1 - e^{-2n/(110^2)}$ .

Finally, recall that we needed  $\varepsilon' = \varepsilon + 1/2^n \leq 1/42000$ . Thus we can take  $\varepsilon \leq 1/43000$  (for large enough  $n$ ). So, if it's possible to pseudodeterministically estimate  $H(\Pr[C = 1])$  within  $\frac{1}{43000}$  with probability  $1 - \delta$ , it's possible to pseudodeterministically estimate  $\Pr[C = 1]$  within  $\frac{1}{100}$  with probability  $1 - \delta - e^{-2n/(110^2)}$ .  $\blacktriangleleft$

# Online Paging with a Vanishing Regret

**Yuval Emek**

Faculty of Industrial Engineering and Management,  
Technion – Israel Institute of Technology, Haifa, Israel  
yemek@technion.ac.il

**Shay Kutten**

Faculty of Industrial Engineering and Management,  
Technion – Israel Institute of Technology, Haifa, Israel  
kuttent@technion.ac.il

**Yangguang Shi**

Faculty of Industrial Engineering and Management,  
Technion – Israel Institute of Technology, Haifa, Israel  
shiyangguang@campus.technion.ac.il

---

## Abstract

This paper considers a variant of the online *paging* problem, where the online algorithm has access to multiple *predictors*, each producing a sequence of predictions for the page arrival times. The predictors may have occasional prediction errors and it is assumed that at least one of them makes a sublinear number of prediction errors in total. Our main result states that this assumption suffices for the design of a randomized online algorithm whose time-average *regret* with respect to the optimal offline algorithm tends to zero as the time tends to infinity. This holds (with different regret bounds) for both the *full information* access model, where in each round, the online algorithm gets the predictions of all predictors, and the *bandit* access model, where in each round, the online algorithm queries a single predictor.

While online algorithms that exploit inaccurate predictions have been a topic of growing interest in the last few years, to the best of our knowledge, this is the first paper that studies this topic in the context of multiple predictors for an online problem with unbounded request sequences. Moreover, to the best of our knowledge, this is also the first paper that aims for (and achieves) online algorithms with a vanishing regret for a classic online problem under reasonable assumptions.

**2012 ACM Subject Classification** Theory of computation → Online learning algorithms; Theory of computation → Caching and paging algorithms

**Keywords and phrases** online paging, inaccurate predictions, multiple predictors, vanishing regret, full information vs. bandit access

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.67

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/2011.09439>.

**Funding** *Yuval Emek*: The work of Yuval Emek was supported in part by an Israeli Science Foundation grant number 1016/17.

*Shay Kutten*: The work of Shay Kutten was supported in part by a grant from the ministry of science in the program that is joint with JSPS and in part by the BSF.

*Yangguang Shi*: The work of Yangguang Shi was partially supported at the Technion by a fellowship of the Israel Council for Higher Education.

## 1 Introduction

A critical bottleneck in the performance of digital computers, known as the “memory wall”, is that the main memory (a.k.a. DRAM) is several orders of magnitude slower than the multiprocessor [26, 16, 4]. Modern computer architectures bridge this performance gap by utilizing a cache, namely, a memory structure positioned next to the multiprocessor that



© Yuval Emek, Shay Kutten, and Yangguang Shi;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 67; pp. 67:1–67:20



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

responds much faster than the main memory. However, the cache is inherently smaller than the main memory which means that some of the memory items requested by the running program may be missing from the cache. When such a *cache miss* occurs, the multiprocessor is required to fetch the requested item from the main memory into the cache; if the cache is already full, then some previously stored item must be evicted to make room for the new one. Minimizing the number of cache misses is known to be a primary criterion for improving the computer's performance [26, 17].

The aforementioned challenge is formalized by means of a classic *online* problem called *paging* [21] (a.k.a. unweighted *caching*), defined over a main memory that consists of  $n \in \mathbb{Z}_{>0}$  *pages* and a cache that holds  $k \in \mathbb{Z}_{>0}$  pages at any given time,  $k < n$ . The execution of a paging algorithm **Alg** progresses in  $T \in \mathbb{Z}_{>0}$  discrete *rounds*, where round  $t \in T$  occupies the time interval  $[t, t-1)$ . An instance of the paging problem is given by a sequence  $\sigma = \{\sigma_t\}_{t \in [T]}$  of page *requests* so that request  $\sigma_t \in [n]$  is revealed at time  $t \in [T]$ . Denoting the *cache configuration* of **Alg** at time  $t$  by  $C_t \subset [n]$ ,  $|C_t| = k$ , if  $\sigma_t \in C_t$ , then **Alg** does nothing in round  $t$ ; otherwise ( $\sigma_t \notin C_t$ ), a *cache miss* occurs and **Alg** should bring the requested page into the cache so that  $\sigma_t \in C_{t+1}$ . Since  $|C_{t+1}| = |C_t| = k$ , it follows that upon a cache miss, **Alg** must evict some page  $i \in C_t$  and its policy is reduced to the selection of this page  $i$ . The *cost* incurred by **Alg** on  $\sigma$  is defined to be the number of cache misses it suffers throughout the execution, denoted by

$$\text{cost}_\sigma(\text{Alg}) = |\{t \in [T] : \sigma_t \notin C_t\}| ,$$

taking the expectation if **Alg** is a randomized algorithm. When  $\sigma$  is clear from the context, we may omit the subscript, writing  $\text{cost}(\text{Alg}) = \text{cost}_\sigma(\text{Alg})$ .

**NAT and the FitF Algorithm.** To avoid cumbersome notation, we assume hereafter that the request sequence  $\sigma$  is augmented with a suffix of  $n$  virtual requests so that  $\sigma_{T+i} = i$  for every  $i \in [n]$ . This facilitates the definition of the *next arrival time (NAT)* of page  $i \in [n]$  with respect to time  $t \in [T]$  as the first time after  $t$  at which page  $i$  is requested, denoted by

$$A_t(i) = \min\{t' > t \mid \sigma_{t'} = i\} .$$

Based on that, we can define the **FitF** (stands for furthest in the future) paging algorithm that on a cache miss at time  $t \in [T]$ , evicts the page  $i \in C_t$  that maximizes  $A_t(i)$ . A classic result of Belady [5] states that **FitF** is optimal in terms of the cost it incurs for the given request sequence  $\sigma$ ; we subsequently denote  $\text{OPT}_\sigma = \text{cost}_\sigma(\text{FitF})$  and omit the subscript, writing  $\text{OPT} = \text{OPT}_\sigma$ , when  $\sigma$  is clear from the context. It is important to point out that **FitF** is an *offline* algorithm as online algorithms are oblivious to the NATs.

**Regret.** We define the *regret* of an online paging algorithm **Alg** on  $\sigma$  as

$$\text{regret}_\sigma(\text{Alg}) = \text{cost}_\sigma(\text{Alg}) - \text{OPT}_\sigma$$

and omit the subscript, writing  $\text{regret}(\text{Alg}) = \text{regret}_\sigma(\text{Alg})$  when  $\sigma$  is clear from the context. Our goal in this paper is to develop an online algorithm that admits a *vanishing regret*, namely, an online algorithm **Alg** for which it is guaranteed that

$$\lim_{T \rightarrow \infty} \frac{\sup \{\text{regret}_\sigma(\text{Alg}) \mid \sigma \in [n]^T\}}{T} = 0 .$$

The following theorem states that this goal is hopeless unless the online algorithm has access to some additional information; its proof should be a folklore, we add it in the full version [11] of this paper for completeness.

► **Theorem 1.** Fix  $n = k + 1$  and let  $\sigma$  be a request sequence generated by picking  $\sigma_t$  uniformly at random (and independently) from  $[n]$  for  $t = 1, \dots, T$ . Then,  $\mathbb{E}(\text{cost}(\text{Alg})) \geq \Omega\left(\frac{T}{k}\right)$  for any (possibly randomized) online paging algorithm  $\text{Alg}$ , whereas  $\mathbb{E}(\text{OPT}) \leq O\left(\frac{T}{k \log k}\right)$ .

## 1.1 Machine Learned Predictions

Developments in *machine learning (ML)* technology suggest a new direction for reducing the number of cache misses by means of predicting the request sequence. Indeed, recent studies have shown that neural networks can be employed to predict the memory pages accessed by a program with high accuracy [16, 9, 22, 19, 23]. When provided with an accurate prediction of the request sequence  $\sigma$ , one can simply simulate **FitF**, thus ensuring an optimal performance.

Unfortunately, the predictions generated by ML techniques are usually not 100% accurate as a result of a distribution drift between the training and test examples or due to adversarial examples [24, 18]. This gives rise to a growing interest in developing algorithmic techniques that can overcome inaccurate predictions, aiming for the design of online algorithms with performance guarantee that improves as the predictions become more accurate [18, 20, 1, 25]. The existing literature in this line of research studies a setting where the online algorithm  $\text{Alg}$  is provided with a sequence of predictions for  $\sigma$  and focuses on bounding  $\text{Alg}$ 's *competitive ratio* as a function of the proximity of this sequence to  $\sigma$  (more on that in Section 2).

The current paper tackles the challenge of overcoming inaccurate predictions from a different angle: Motivated by the abundance of forecasting algorithms that may be trained on different data sets or using different models (e.g., models that are robust to adversarial examples [15]), we consider a decision maker with access to *multiple* predicting sequences for  $\sigma$ . Our main goal is to design an online algorithm  $\text{Alg}$  that admits a vanishing regret assuming that at least one of the predicting sequences is sufficiently accurate, even though the decision maker does not know in advance which predicting sequence it is.

**Explicit Predictors.** Formally, we consider  $M \in \mathbb{Z}_{>0}$  predictors whose role is to predict the request sequence  $\sigma$ . In the most basic form, referred to hereafter as the *explicit predictors* setting, each predictor  $j \in [M]$  produces a page sequence  $\pi^j = \{\pi_t^j\}_{t \in [T]} \in [n]^T$ , where  $\pi_t^j$  aims to predict  $\sigma_t$  for every  $t \in [T]$ , and the sequences  $\pi^1, \dots, \pi^M$  are revealed to the online algorithm  $\text{Alg}$  at the beginning of the execution. Under the explicit predictors setting, predictor  $j \in [M]$  is said to have a *prediction error* in round  $t \in [T]$  if  $\pi_t^j \neq \sigma_t$ . We measure the accuracy of predictor  $j$  by means of her *cumulative prediction error*

$$\eta_e^j = \eta_e(\pi^j) = \left| \left\{ t \in [T] : \pi_t^j \neq \sigma_t \right\} \right|$$

and define  $\eta_e^{\min} = \min\{\eta_e^j \mid j \in [M]\}$ .

The fundamental assumption that guides the current paper, referred to hereafter as the *good predictor* assumption, is that there exists at least one predictor whose cumulative prediction error is sublinear in  $T$ , namely,  $\eta_e^{\min} = o(T)$ . We emphasize that  $\text{Alg}$  has no a priori knowledge of  $\eta_e^1, \dots, \eta_e^M$  nor does it know the predictor that realizes  $\eta_e^{\min}$ . Our main research question can now be stated as follows:

Does the good predictor assumption provide a sufficient condition for the existence of an online algorithm that admits a vanishing regret?

**NAT Predictors.** For the paging problem, it is arguably more natural to consider the setting of *NAT predictors*, where predictor  $j \in [M]$  produces in each round  $t \in [T]$ , a prediction  $a_t^j \in (t, T + n]$  for the NAT  $A_t(\sigma_t)$  of the page that has just been requested. Under this

setting, predictor  $j \in [M]$  is said to have a *prediction error* in round  $t \in [T]$  if  $a_t^j \neq A_t(\sigma_t)$ . As in the explicit predictors setting, we measure the accuracy of (NAT) predictor  $j$  by means of her *cumulative prediction error*, now defined as

$$\eta_N^j = \left| \left\{ t \in [T] : a_t^j \neq A_t(\sigma_t) \right\} \right| \quad (1)$$

(this measure is termed *classification loss* in [18]), and define  $\eta_N^{\min} = \min\{\eta_N^j \mid j \in [M]\}$ .<sup>1</sup> The NAT predictors version of the good predictor assumption states that  $\eta_N^{\min} = o(T)$ .

Given a page sequence  $\pi = \{\pi_t\}_{t \in [T]} \in [n]^T$  augmented with a suffix of  $n$  pages such that  $\pi_{T+i} = i$  for every  $i \in [n]$ , we say that (NAT) predictor  $j \in [M]$  is *consistent* with  $\pi$  if  $a_t^j = \min\{t' > t \mid \pi_{t'} = \sigma_t\}$  for every  $t \in [T]$ ; if the page sequence  $\pi$  is not important or clear from the context, then we may say that predictor  $j$  is *consistent* without mentioning  $\pi$ . The key observation here is that if predictor  $j$  is consistent with a page sequence  $\pi$ , then  $\eta_N^j$  provides a good approximation for  $\eta_e(\pi)$ , specifically,

$$\eta_e(\pi) - n \leq \eta_N^j \leq 2 \cdot \eta_e(\pi) \quad (2)$$

(the proof is deferred to the full version [11]). This means that the setting of NAT predictors is stronger than that of explicit predictors in the sense that NAT predictor  $j \in [M]$  can be simulated (consistently) from explicit predictor  $j$  by deriving the NAT prediction  $a_t^j$  in round  $t \in [T]$  from the (explicit) predictions  $\pi_{t+1}^j, \pi_{t+2}^j, \dots, \pi_T^j$ , while ensuring that  $\eta_N^j$  is a good approximation for  $\eta_e^j$ . Therefore, unless stated otherwise, we subsequently restrict our attention to NAT predictors and in particular omit the subscript from the cumulative prediction error notation, writing  $\eta^j = \eta_N^j$  and  $\eta^{\min} = \eta_N^{\min}$ . It is important to point out though that the results established in the current paper hold regardless of whether the (NAT) predictors are consistent or not.

**Access Models.** Recall that the (NAT) predictors  $j \in [M]$  produce their predictions in an online fashion so that the NAT prediction  $a_t^j$  is produced in round  $t$ . This calls for a distinction between two *access models* that determine the exact manner in which  $a_t^j$  is revealed to the online paging algorithm **Alg**. First, we consider the *full information* access model, where in each round  $t \in [T]$ , **Alg** receives  $a_t^j$  for all  $j \in [M]$ . Motivated by systems in which accessing the ML predictions is costly in both time and space (thus preventing **Alg** from querying multiple predictors in the same round and/or predictions belonging to past rounds), we also consider the *bandit* access model, where in each round  $t \in [T]$ , **Alg** receives  $a_t^j$  for a single predictor  $j \in [M]$  selected by **Alg** in that round. To make things precise, we assume, under both access models, that if **Alg** has to evict a page in round  $t$ , then the decision on the evicted page is made prior to receiving the prediction(s) in that round. Notice though that the information that **Alg** receives from the predictor(s) is not related to the evicted page and as such, should not be viewed as a feedback that **Alg** receives in response to the action it takes in the current round.

## 1.2 Our Contribution

Consider a (single) predictor that in each round  $t \in [T]$ , produces a prediction  $a_t$  for the NAT  $A_t(\sigma_t)$  of the page that has just been requested and let  $\eta$  be her cumulative prediction error. Our first technical contribution comes in the form of a thorough analysis of the performance

<sup>1</sup> In Section 2, we provide a refined definition for the cumulative prediction error of a NAT predictor that is more robust against adversarial interference such as shifting each  $a_t^j$  by a constant. For simplicity of the exposition, the definition presented in Eq. (1) is used throughout the current section; we emphasize though that all our results hold for the stronger notion of prediction error as defined in Section 2.



of a simple online paging algorithm called **Sim** that simulates **FitF**, replacing the actual NATs with the ones derived from the prediction sequence  $\{a_t\}_{t \in [T]}$ . Using some careful combinatorial arguments, we establish the following bound.

► **Theorem 2.** *The regret of **Sim** satisfies  $\text{regret}(\text{Sim}) \leq O(\eta + k)$ .*

Relying on online learning techniques, Blum and Burch [6] develop an online algorithm “multiplexer” that given multiple online algorithms as subroutines, produces a randomized online algorithm that performs almost as good as the best subroutine in hindsight. Applying Theorem 2 to the  $M$  predictors so that each predictor  $j \in [M]$  yields its own online paging algorithm  $\text{Sim}^j$  and plugging algorithms  $\text{Sim}^1, \dots, \text{Sim}^M$  into the multiplexer of [6], we establish the following theorem, thus concluding that the good predictor assumption implies an online paging algorithm with a vanishing regret under the full information access model.

► **Theorem 3.** *There exists a randomized online paging algorithm that given full information access to  $M$  NAT predictors with minimum cumulative prediction error  $\eta^{\min}$ , has regret at most  $O(\eta^{\min} + k + (Tk \log M)^{1/2})$ .*

Combined with (2), we obtain the same asymptotic regret bound for explicit predictors.

► **Corollary 4.** *There exists a randomized online paging algorithm that given access to  $M$  explicit predictors with minimum cumulative prediction error  $\eta_e^{\min}$ , has regret at most  $O(\eta_e^{\min} + k + (Tk \log M)^{1/2})$ .*

The explicit predictors setting is general enough to make it applicable to virtually any online problem. This raises the question of whether other online problems admit online algorithms with a vanishing regret given access to explicit predictors whose minimum cumulative prediction error is sublinear in  $T$ . We view the investigation of this question as an interesting research thread that will hopefully arise from the current paper.

Going back to the setting of NAT predictors, one wonders if a vanishing regret can be achieved also under the bandit access model since the technique of [6] unfortunately does not apply to this more restricted access model. An inherent difficulty in the bandit access model is that we cannot keep track of the cache configuration of  $\text{Sim}^j$  unless predictor  $j$  is queried in each round (which means that no other predictor can be queried). To overcome this obstacle, we exploit certain combinatorial properties of the **Sim** algorithm to show that  $\text{Sim}^j$  can be “chased” without knowing its current cache configuration, while bounding the accumulated cost difference. By a careful application of online learning techniques, this allows us to establish the following theorem, thus concluding that the good predictor assumption implies an online paging algorithm with a vanishing regret under the bandit access model as well.

► **Theorem 5.** *There exists a randomized online paging algorithm that given bandit access to  $M$  NAT predictors with minimum cumulative prediction error  $\eta^{\min}$ , has regret at most  $O(\eta^{\min} + T^{2/3}kM^{1/2})$ .*

### 1.3 Related Work and Discussion

We say that an online algorithm **Alg** for a minimization problem  $\mathcal{P}$  has *competitive ratio*  $\alpha$  if for any instance  $\sigma$  of  $\mathcal{P}$ , the cost incurred by **Alg** on  $\sigma$  is at most  $\alpha \cdot \text{OPT}_\sigma + \beta$ , where  $\text{OPT}_\sigma$  is the cost incurred by an optimal offline algorithm on  $\sigma$  and  $\beta$  is a constant that may depend on  $\mathcal{P}$ , but not on  $\sigma$  [21, 7]. In comparison, the notion of regret as defined in the current paper uses the optimal offline algorithm as an absolute (additive), rather than

relative (multiplicative), benchmark. Notice that the vanishing regret condition cannot be expressed in the scope of the competitive ratio definition. In particular,  $\alpha = 1$  is a stronger requirement than vanishing regret as the latter can accommodate an additive parameter  $\beta$  that does depend on  $\sigma$  as long as it is sublinear in  $T = |\sigma|$ . On the other hand,  $\alpha > 1$  implies a non-vanishing regret when  $\text{OPT}_\sigma$  scales linearly with  $T$ .

As mentioned in Section 1.1, most of the existing literature on augmenting online algorithms with ML predictions is restricted to the case of a single predictor [18, 20, 1, 25]. The goal of these papers is to develop online algorithms with two guarantees: (i) their competitive ratio tends to  $O(1)$  (though not necessarily to 1) as the predictor's accuracy improves; and (ii) they are robust in the sense that regardless of the predictor's accuracy, their competitive ratio is not much worse than that of the best online algorithm that has no access to predictions.

In contrast, the current paper addresses the setting of multiple predictors, working under the assumption that at least one of them is sufficiently accurate, and seeking to develop online algorithms with a vanishing regret. To the best of our knowledge, this is the first paper that aims at this direction.

Most closely related to the current paper are the papers of [18, 20, 25] on online paging with predictions. The authors of these papers stick to the setting of a (single) NAT predictor and quantify the predictor's accuracy by means of the  $L_1$ -norm. Specifically, taking  $\{a_t\}_{t \in [T]}$  to be the sequence of NAT predictions, they define the predictor's cumulative prediction error to be  $\sharp L_1 = \sum_t |a_t - A_t(\sigma_t)|$ . It is easy to see that for any NAT predictor, the cumulative prediction error as defined in (1) is never larger than its  $\sharp L_1$ , while the former can be  $\Omega(T)$ -times smaller.

In particular, Lykouris and Vassilvitskii [18] design a randomized online paging algorithm whose competitive ratio is at most  $O\left(\min\left\{1 + \sqrt{\sharp L_1 / \text{OPT}}, \log k\right\}\right)$ . Rohatgi [20] presents an improved randomized online algorithm with competitive ratio upper bounded by  $O\left(\min\left\{1 + \frac{\log k}{k} \frac{\sharp L_1}{\text{OPT}}, \log k\right\}\right)$  and accompany this with a lower bound of  $\Omega\left(\min\left\{1 + \frac{1}{k \log k} \frac{\sharp L_1}{\text{OPT}}, \log k\right\}\right)$ . Notice that the online algorithms presented in [18, 20] belong to the *marking family* of paging algorithms [13] and it can be shown that the competitive ratio of any such algorithm is bounded away from 1 even when provided with a fully accurate predictor (consider for example the paging instance defined by setting  $n = 4$ ,  $k = 2$ , and  $\sigma_t = (t \bmod 4) + 1$  for every  $t \in [T]$ ).

Recently, Wei [25] advanced the state of the art of this problem further, presenting a randomized  $O\left(\min\left\{1 + \frac{1}{k} \frac{\sharp L_1}{\text{OPT}}, \log k\right\}\right)$ -competitive online paging algorithm. To do so, Wei analyzes an algorithm called *BlindOracle*, that can be viewed as a variant of our *Sim* algorithm (see Section 1.2), and proves that its competitive ratio is at most  $\min\left\{1 + O\left(\frac{\sharp L_1}{\text{OPT}}\right), O\left(1 + \frac{1}{k} \frac{\sharp L_1}{\text{OPT}}\right)\right\}$ . He then plugs this algorithm into the multiplexer of [6] together with an  $O(\log k)$ -competitive off-the-shelf randomized online paging algorithm to obtain the promised competitive ratio. Notice that the bound that Wei establishes on the competitive ratio of *BlindOracle* immediately implies an  $O(\sharp L_1)$  bound on the regret of this algorithm. As such, Theorem 2 can be viewed as a refinement of Wei's result, bounding the regret as a function of  $\eta$  rather than the weaker measure of  $\sharp L_1$ .

Antoniadis et al. [1] studies online algorithms with ML predictions in the context of the *metrical task system (MTS)* problem [8]. They consider a different type of predictor that in each round  $t \in [T]$ , provides a prediction  $\hat{s}_t$  for the state  $s_t$  of an optimal offline algorithm, measuring the prediction error by means of  $\sharp \text{Distances} = \sum_{t \in [T]} \text{dist}(s_t, \hat{s}_t)$ , where  $\text{dist}(\cdot, \cdot)$  is the distance function of the underlying metric space. It is well known that any paging

instance  $\mathcal{I}$  can be transformed into an MTS instance  $\mathcal{I}_{\text{MTS}}$ . Antoniadis et al. prove that the prediction sequence  $\{a_t\}_{t \in [T]}$  of a NAT predictor for  $\mathcal{I}$  can also be transformed into a prediction sequence  $\{\hat{s}_t\}_{t \in [T]}$  for  $\mathcal{I}_{\text{MTS}}$ . However, the resulting prediction error  $\# \text{Distances}$  of the latter sequence is incomparable to the prediction error  $\# \text{L}_1$  of the former; this remains true also for the stronger notion of cumulative prediction error as defined in (1).

Online algorithms with access to multiple predictors have been studied by Gollapudi and Panigrahi for the *ski rental* problem [14]. Among other results, they prove that a competitive ratio of  $\alpha = \frac{4}{3}$  (resp.,  $\alpha = \frac{\sqrt{5}+1}{2}$ ) can be achieved by a randomized (resp., deterministic) online algorithm that has access to two predictors assuming that at least one of them provides an accurate prediction for the number of skiing days. Notice that the length of the request sequence in the ski rental problem is inherently bounded by the cost of buying the ski gear; this is in contrast to the paging problem considered in the current paper, where much of the challenge comes from the unbounded request sequence.

The reader may have noticed that some of the terminology used in the current paper is borrowed from the *online learning* domain [10]. The main reason for this choice is that the research objectives of the current paper are, to a large extent, more in line with the objectives common to the online learning literature than they are in line with the objectives of the literature on online computation. In particular, as discussed already, we measure the quality of our online algorithms by means of their regret (rather than competitiveness), indicating that the online algorithm can be viewed as a decision maker that tries to learn the best offline algorithm.

## 1.4 Paper's Organization

The remainder of this paper is organized as follows. In Section 2, we refine the notion of cumulative prediction error as defined in (1) and compare the refined notion with the number of inversions used in some of the related literature [20, 25]. The analysis of the **Sim** algorithm (using a single predictor), leading to the proof of Theorem 2, is carried out in Section 3. Section 4 is then dedicated to the setting of multiple (NAT) predictors under the bandit access model and establishes Theorem 5. As discussed in Section 1.3, Theorem 3, dealing with the full information access model, follows from Theorem 2 combined with a technique of [6]; this is explained in more detail in the full version [11].

## 2 Measurements of Prediction Errors

The measurement of the prediction errors plays an important role in the study on online paging algorithms augmented by predictions. This part makes a comparison between different measurements for the scenario where there is a single NAT predictor. To avoid ambiguity, in this part we use  $\# \text{ErrorRounds}$  to represent the measurement defined in Eq. (1) for the single predictor  $j$ . In the following, the superscript  $j$  for the predictor is omitted for convenience.

In the analysis of [25], the prediction errors are measured with the number of *inverted pairs*. For a pair of two rounds  $\{t, t'\}$ , we say it is an inverted pair if  $A_t(\sigma_t) < A_{t'}(\sigma_{t'})$  and  $a_t \geq a_{t'}$ . Let  $\text{INV}$  be the set of all the inverted pairs, and define  $\# \text{InvertedPairs} = |\text{INV}|$ . To compare the measurement  $\# \text{InvertedPairs}$  with  $\# \text{ErrorRounds}$ , we also define the following notations.

$$\begin{aligned} \# \text{InvertedRounds} &\doteq \left| \{t \mid \exists t' \text{ s.t. } \{t, t'\} \in \text{INV}\} \right| \\ \# \text{ErrorRoundsInInversion} &\doteq \left| \{t \mid A_t(\sigma_t) \neq a_t \wedge \exists t' \text{ s.t. } \{t, t'\} \in \text{INV}\} \right| \end{aligned}$$

First, it trivially holds that  $\# \text{InvertedRounds} \leq 2 \cdot \# \text{InvertedPairs}$ , and  $\# \text{InvertedPairs}$  can be  $\Omega(T)$  times larger than  $\# \text{InvertedRounds}$ . To see the second claim, consider the following sequence  $\sigma$  of requested pages and a prediction sequence  $\pi$ .

$$\sigma_t = \begin{cases} 1 & \text{if } t \leq \frac{T}{2} \\ 2 & \text{if } t > \frac{T}{2} \end{cases}, \quad \text{and} \quad \pi_t = \begin{cases} 2 & \text{if } t \leq \frac{T}{2} \\ 1 & \text{if } t > \frac{T}{2} \end{cases}.$$

It can be verified that for a sequence of predictions in the form of NATs that are consistent with the settings above,  $\# \text{InvertedPairs}$  is in the order of  $T^2$  while  $\# \text{InvertedRounds}$  is in the order of  $T$ .

Second, we claim that  $\# \text{InvertedRounds}$  and  $\# \text{ErrorRounds}$  are incomparable, which means that there exists an example where  $\# \text{InvertedRounds}$  is  $\Omega(T)$  times larger than  $\# \text{ErrorRounds}$ , and vice versa. Still, we demonstrate these examples with the sequence  $\sigma$  of requests and the prediction sequence  $\pi$ , and the claims above can be verified after converting the predictions in the form of requests to consistent predictions in the form of NATs. The configuration of the first example is given as follows.

$$\sigma_t = \begin{cases} (t \bmod (k-1)) + 1 & \text{if } 1 < t < T \\ k & \text{otherwise} \end{cases}, \quad \text{and} \quad \pi = \begin{cases} (t \bmod (k-1)) + 1 & \text{if } t > 2 \\ k & \text{otherwise} \end{cases}.$$

The second example is configured as follows.

$$\sigma_t = (t \bmod (k-1)) + 1, \quad \text{and} \quad \pi = \begin{cases} ((t-1) \bmod (k-1)) + 1 & \text{if } t > 1 \\ k & \text{otherwise} \end{cases}.$$

Third, it is obvious that

$$\# \text{ErrorRoundsInInversion} \leq \min \{ \# \text{ErrorRounds}, \# \text{InvertedRounds} \}.$$

Although in Section 1.1 we define  $\eta^j$  for every predictor  $j$  in the form of  $\# \text{ErrorRounds}$  for simplicity, our technique indeed works for the better measurement  $\# \text{ErrorRoundsInInversion}$ . Therefore, in the technical parts of the current paper, including Section 3 and Section 4, we use the following refined definition of  $\eta^j$  by abuse of notation:

$$\eta^j \doteq \left| \{t \mid A_t(\sigma_t) \neq a_t^j \wedge \exists t' \text{ s.t. } \{t, t'\} \in \text{INV}^j\} \right|,$$

where  $\text{INV}^j = \{(t, t') \mid A_t(\sigma_t) < A_{t'}(\sigma_{t'}) \wedge a_t^j \geq a_{t'}^j\}$ .

### 3 Single NAT Predictor

We start with the NAT predictor setting with  $M = 1$ . In such a case, there is no difference between the full information access model and the bandit access model. Throughout this section, we still omit the superscript  $j$  for the index of the predictor.

The algorithm **Sim** that we consider for this setting simulates **FitF** with maintaining a value  $\hat{a}_t(i)$ , which we call the *remedy prediction*, for each round  $t \in [T]$  and each page  $i \in [n]$ . In particular, for each page  $i \in [n]$ , **Sim** sets

$$\begin{aligned} \hat{a}_1(i) &= \begin{cases} a_1 & \text{if } i = \sigma_1 \\ Z+1 & \text{otherwise} \end{cases}, \quad \text{and} \\ \forall t \in [2, T] \quad \hat{a}_t(i) &= \begin{cases} a_t & \text{if } i = \sigma_t \\ Z & \text{if } \hat{a}_{t-1}(i) \leq t \wedge i \neq \sigma_t \wedge \hat{a}_{t-1}(i) \leq \hat{a}_{t-1}(\sigma_t) < Z, \\ \hat{a}_{t-1}(i) & \text{otherwise} \end{cases} \end{aligned} \quad (3)$$

where  $Z > T + n$  is a sufficiently large integer. For each round  $t$  when a cache miss happens, the algorithm evicts the page  $\hat{e}_t = i$  that maximizes  $\hat{a}_t(i)$ , and ties are broken in an arbitrary way. The following statements can be directly inferred from Eq. (3).

- **Lemma 6.** *The following properties are satisfied for every round  $t \in [T]$ :*
- *If  $a_t > A_t(\sigma_t)$ , then for each round  $t' \in [t, \min\{T, A_t(\sigma_t) - 1\}]$ , we have  $\hat{a}_{t'}(\sigma_t) > A_{t'}(\sigma_t)$ .*
  - *If  $a_t < A_t(\sigma_t)$ , then for each round  $t' \in [t, \min\{T, A_t(\sigma_t) - 1\}]$ , either  $\hat{a}_{t'}(\sigma_t) < A_{t'}(\sigma_t)$  or  $\hat{a}_{t'}(\sigma_t) = Z > A_{t'}(\sigma_t)$ . Particularly, if  $t' \leq a_t - 1$ , then  $\hat{a}_{t'}(\sigma_t) < A_{t'}(\sigma_t)$ .*
  - *If  $a_t = A_t(\sigma_t)$ , then for each round  $t' \in [t, \min\{T, A_t(\sigma_t) - 1\}]$ , we have  $\hat{a}_{t'}(\sigma_t) = A_{t'}(\sigma_t)$ .*

Next, we will analyze the cost incurred by **Sim** and show that it has a vanishing regret.

### 3.1 Definitions and Notations for Analysis

For each round  $t$ , we use  $e_t$  and  $\hat{e}_t$  to represent the pages that are evicted by **FitF** and **Sim**, respectively. We say  $e_t = \perp$  (resp.  $\hat{e}_t = \perp$ ) if **FitF** (resp. **Sim**) does not evict any page.

For each page  $i \in [n]$  and each round  $t \in [T]$ , define  $R_t(i)$  to be the last round before  $t$  when  $i$  is requested. Formally,

$$R_t(i) \doteq \begin{cases} \max\{t' < t \mid \sigma_{t'} = i\} & \text{if } \exists t' \in [1, t) \text{ s.t. } \sigma_{t'} = i \\ -1 & \text{otherwise} \end{cases}. \quad (4)$$

The following results can be inferred from Eq. (4) and Lemma 6.

For any round  $t \in [T]$ , let  $C_t$  and  $\hat{C}_t$  be the *cache profiles* incurred by **FitF** and **Sim**, respectively. More specifically,  $C_1$  and  $\hat{C}_1$  represent the cache items given at the beginning. To provide tools for the more complicated scenario where there are multiple predictors, the analysis in this section is carried out without assuming that  $C_1 = \hat{C}_1$ . For each  $t \in [T-1]$ , the cache profile  $C_t$  (resp.  $\hat{C}_t$ ) is updated to  $C_{t+1}$  (resp.  $\hat{C}_{t+1}$ ) immediately after **FitF** (resp. **Sim**) has processed the request  $\sigma_t$ . The cache profiles of **FitF** and **Sim** after serving  $\sigma_T$  are denoted by  $C_{T+1}$  and  $\hat{C}_{T+1}$ , respectively.

Denote the intersection between the cache profiles at each round  $t \in [T]$  by  $I_t = C_t \cap \hat{C}_t$ . Define the *distance* between the cache profiles to be  $d_t = k - |I_t|$ . We use  $\delta_t$  to represent the difference in the costs between **FitF** and **Sim** for serving  $\sigma_t$ . Formally,  $\delta_t \doteq 1_{\sigma_t \notin \hat{C}_t} - 1_{\sigma_t \notin C_t}$ .

Define  $\mathcal{H}_x^y$  for  $x \in \mathbb{Z}$ ,  $y \in \mathbb{Z}$  to be the set of rounds  $t \in [T]$  where the  $d_{t+1} - d_t = x$  and  $\delta_t = y$ . Let  $\mathcal{H}_x = \bigcup_y \mathcal{H}_x^y$  and  $\mathcal{H}^y = \bigcup_x \mathcal{H}_x^y$ .

For a round  $t \in [T]$ , we say that  $t$  is a *troublemaker* if and only if  $t$  satisfies

$$(\hat{e}_t \neq \perp) \wedge (\hat{e}_t \in I_t) \wedge A_t(\hat{e}_t) \in [T] \wedge (\hat{e}_t \in C_{A_t(\hat{e}_t)}) \wedge (e_t = \perp \vee e_t \notin I_t). \quad (5)$$

The set of troublemaker rounds is denoted by  $\Gamma$ . For any troublemaker  $\gamma \in \Gamma$  and any round  $t \in (\gamma, T]$ , we say  $\gamma$  is *active* at  $t$  if  $t < A_\gamma(\hat{e}_\gamma)$ . The set of troublemakers that are active at  $t$  is denoted by  $\Gamma_t \subseteq \Gamma \cap [t-1]$ . The *active period*  $[\gamma+1, A_\gamma(\hat{e}_\gamma)-1]$  of  $\gamma$  is denoted by  $\theta_\gamma$ .

**Preliminary results.** The following results are directly inferred from the definitions above.

- **Lemma 7.** *For any round  $t$  and any page  $i$ , the following properties are satisfied.*

- *If  $R_t(i) = -1$  and  $i \neq \sigma_t$ , then  $\hat{a}_t(i) = Z + 1$ , and vice versa.*
- *The equality  $\hat{a}_t(i) = A_t(i)$  holds if  $a_r = A_r(\sigma_r)$ , where  $r = R_t(i)$ .*

**Proof.** The first statement can be proved inductively with using Eq. (3). The first statement implies that if  $\hat{a}_t(i) = A_t(i)$ , then  $R_t(i) \neq -1$ . Therefore, the second statement can be inferred from Lemma 6. ◀

► **Lemma 8.**  $|\{t \mid \hat{e}_t \neq \perp \wedge R_t(\hat{e}_t) = -1\}| \leq k$ .

**Proof.** For each round  $t$  and each page  $i \in \hat{C}_t$ , the equality  $R_t(i) = -1$  holds only if  $i \in \hat{C}_1$ . The initial cache profile  $\hat{C}_1$  contains  $k$  different pages, and for each page  $i \in \hat{C}_1$ , if there exist two rounds  $t, t'$  with  $t < t'$  and  $\hat{e}_t = \hat{e}_{t'} = i$ , then  $R_{t'}(i) \geq t \neq -1$ . This implies that there are at most  $k$  rounds  $t$  with  $R_t(\hat{e}_t) = -1$ . ◀

► **Lemma 9.**  $|\{t \mid R_t(\sigma_t) = -1 \wedge t \in \mathcal{H}_0^1\}| \leq k$

**Proof.** For a round  $t \in \mathcal{H}_0^1$ , we have  $\sigma_t \in C_t$ . Since  $R_t(\sigma_t) = -1$ , we have  $\sigma_t \in C_1$ . Then this lemma can be proved in a similar way with Lemma 8. ◀

► **Lemma 10.** For every round  $t$  and any troublemaker  $\gamma \in \Gamma_t$ , we have (1)  $\hat{e}_\gamma \in C_t \setminus \hat{C}_t$ , and (2)  $\hat{e}_\gamma \neq \hat{e}_{\gamma'}$  for any troublemaker  $\gamma' \in \Gamma_t$  with  $\gamma \neq \gamma'$ .

**Proof.** The first statement is directly inferred from the definition of active troublemakers. Now consider the second statement. Without loss of generality, we assume that  $\gamma < \gamma'$ . Then the first statement shows that  $\hat{e}_\gamma \notin \hat{C}_{\gamma'}$ , which means that **Sim** cannot evict  $\hat{e}_\gamma$  at  $\gamma'$ . ◀

### 3.2 Reducing Cost Analysis to Troublemaker Counting

► **Lemma 11.** For each round  $t \in [T]$ , we have  $d_{t+1} - d_t \in \{-1, 0, 1\}$  and  $\delta_t \in \{-1, 0, 1\}$ .

The proof of Lemma 11 is deferred to the full version [11]. This lemma implies that for the sets  $\mathcal{H}_x^y$ , we only need to consider the parameters  $x, y \in \{-1, 0, 1\}$ . The following result on  $\mathcal{H}_x^y$  can be inferred from the proof of Lemma 11.

► **Lemma 12.** It holds that  $\mathcal{H}_1 = \mathcal{H}_1^0 = \{t \mid e_t \neq \perp \wedge \hat{e}_t \neq \perp \wedge e_t \neq \hat{e}_t \wedge \hat{e}_t \in I_t \wedge e_t \in I_t\}$ .

Lemma 12 implies that  $\mathcal{H}_1^1 = \emptyset$ , therefore,  $\text{cost}(\text{Sim}) - \text{OPT}$  can be bounded by  $|\mathcal{H}^1| - |\mathcal{H}^{-1}| = |\mathcal{H}_0^1| + |\mathcal{H}_{-1}^1| - |\mathcal{H}^{-1}|$ .

► **Lemma 13.**  $|\mathcal{H}_{-1}| \leq |\mathcal{H}_1| + k$ .

**Proof.** By definition,  $\mathcal{H}_{-1}$  is the set of rounds  $t$  with  $d_{t+1} - d_t < 0$ , and  $\mathcal{H}_1$  is the set of rounds  $t$  with  $d_{t+1} - d_t > 0$ . Since  $d_1 \leq k$  and  $d_t \geq 0$  for every  $t \in [T + 1]$ , this proposition holds. ◀

Lemma 13 allows us to bound  $|\mathcal{H}_{-1}^1|$  with  $|\mathcal{H}_1|$ . The following result follows from the mechanisms of **FitF** and **Sim** in choosing the page for eviction when cache miss happens.

► **Lemma 14.** For each round  $t \in \mathcal{H}_1$ , we have  $A_t(\hat{e}_t) < A_t(e_t)$  and  $\hat{a}_t(e_t) \leq \hat{a}_t(\hat{e}_t)$ .

For each round  $t \in \mathcal{H}_1$ , we say that  $t$  *blames* another round  $t'$  specified as follows. Let  $r = R_t(\hat{e}_t)$  and  $r' = R_t(e_t)$ , then the round blamed by  $t$  is

$$t' = \begin{cases} r & \text{if } (r \neq -1) \wedge (a_r \neq A_r(\sigma_r)) \\ r' & \text{if } (r = -1 \vee a_r = A_r(\sigma_r)) \wedge (r' \neq -1 \wedge a_{r'} \neq A_{r'}(\sigma_{r'})) \\ -1 & \text{otherwise} \end{cases}.$$

► **Lemma 15.** For each  $t \in \mathcal{H}_1$ , let  $t'$  be the round blamed by  $t$ . If  $t' = -1$ , then  $R_t(\hat{e}_t) = -1$ . If  $R_t(\hat{e}_t) \neq -1$ , then there exists a round  $t''$  such that  $\{t', t''\} \in \text{INV}$ .

**Proof.** For the first claim, suppose on the contrary  $r = R_t(\hat{e}_t) \neq -1$ . Now consider two cases,  $r' \neq -1$  and  $r' = -1$ , where  $r' = R_t(e_t)$ .

- $r' \neq -1$ : Since  $t' = -1$ , in such a case we have  $a_r = A_r(\sigma_r)$  and  $a_{r'} = A_{r'}(\sigma_{r'})$ . By Lemma 7, it holds that  $\hat{a}_t(\hat{e}_t) = a_r = A_r(\sigma_r) = A_t(\hat{e}_t)$ . Similarly, we have  $\hat{a}_t(e_t) = A_t(e_t)$ . This conflicts with Lemma 14.
- $r' = -1$ : By Lemma 7, in such a case we have  $\hat{a}_t(e_t) = Z + 1$ , because  $e_t \neq \sigma_t$ . As it still holds that  $\hat{a}_t(\hat{e}_t) = A_t(\hat{e}_t) < Z$ , we have  $\hat{a}_t(\hat{e}_t) < \hat{a}_t(e_t)$ , which conflicts with Lemma 14.

For the second claim, since  $R_t(\hat{e}_t) \neq -1$ , Lemma 7 indicates that  $\hat{a}_t(\hat{e}_t) \leq Z$ . Now consider the following two cases.

- $\hat{a}_t(\hat{e}_t) < Z$ : By Lemma 14, in such a case we also have  $\hat{a}_t(e_t) < Z$ . It can be inferred from Eq. (3) that  $\hat{a}_t(\hat{e}_t) = a_r$  and  $\hat{a}_t(e_t) = a_{r'}$ . Notice that the second equality holds because  $\hat{a}_t(e_t) < Z$  means that  $r' \neq -1$ . Then by Lemma 14, we have  $a_{r'} = \hat{a}_t(e_t) \leq \hat{a}_t(\hat{e}_t) = a_r$  and  $A_r(\sigma_r) = A_t(\hat{e}_t) < A_t(e_t) = A_{r'}(\sigma_{r'})$ . This means that the pair  $\{r, r'\}$  is an inversion. Since  $R_t(\hat{e}_t) = -1$ , we have either  $t' = r$  or  $t' = r'$ . By taking

$$t'' = \begin{cases} r' & \text{if } t' = r \\ r & \text{if } t' = r' \end{cases},$$

this claim is proved.

- $\hat{a}_t(\hat{e}_t) = Z$ : Let  $t_1$  be the first round in  $(r, t]$  so that  $\hat{a}_{t_1}(\hat{e}_t) = Z$ . Then Eq. (3) indicates that  $\hat{a}_{t_1-1}(\sigma_{t_1}) < Z$ . By Lemma 7, we have  $r_1 = R_{t_1}(\sigma_{t_1}) \neq -1$ . Then,  $A_{r_1}(\sigma_{r_1}) = t_1 < A_t(\hat{e}_t) = A_r(\sigma_r)$ . Moreover,  $\hat{a}_{t_1-1}(\sigma_r) \leq \hat{a}_{t_1-1}(\sigma_{t_1}) < Z$  means that  $a_r = \hat{a}_{t_1-1}(\sigma_r)$  and  $a_{r_1} = \hat{a}_{t_1-1}(\sigma_{t_1})$ . Therefore, the pair  $\{r_1, r\}$  is an inversion. Then this claim is established if  $t' = r$ . This equation holds because (1)  $r = R_t(\hat{e}_t) \neq -1$ , and (2) Lemma 7 implies that  $A_r(\sigma_r) \neq a_r$ , because otherwise  $\hat{a}_t(\hat{e}_t) = A_t(\hat{e}_t) < Z$ .

This completes the proof.  $\blacktriangleleft$

► **Lemma 16.** *For each round  $t' \neq -1$ , it can be blamed by at most two rounds in  $\mathcal{H}_1$ .*

**Proof.** Suppose that  $t'$  is blamed by  $t \in \mathcal{H}_1$  such that  $t' = R_t(e_t)$ . Then for any round  $t'' \in \mathcal{H}_1$  with  $t' < t'' < t$ , it cannot blame  $t'$  by taking  $t' = R_{t''}(e_{t''})$ . This is because if there exists such a round  $t''$ , then by definition we have  $e_t = e_{t''}$ . In such a case, there must exist a round  $\tilde{t} \in (t'', t)$  with  $\sigma_{\tilde{t}} = e_t$ , otherwise  $e_t \notin C_t$ . This conflicts with the definition that  $R_t(e_t)$  is the last round before  $t$  when  $e_t$  is requested. For any  $t'' \in \mathcal{H}_1$  with  $t'' > t$ , it cannot blame  $t'$  by taking  $t' = R_{t''}(e_{t''})$ , either. Still, if  $e_t = e_{t''}$ , there must exist a round  $\tilde{t} \in (t, t'')$  with  $\sigma_{\tilde{t}} = e_t$ . In such a case,  $R_{t''}(e_{t''}) \geq \tilde{t} > t > t'$ . The case  $t' = R_t(\hat{e}_t)$  is symmetric with the case above.  $\blacktriangleleft$

► **Lemma 17.** *It holds that  $|\mathcal{H}_1| \leq 2 \cdot \eta + k$  and  $|\mathcal{H}_{-1}^1| \leq 2 \cdot \eta + 2k$ .*

**Proof.** The statement  $|\mathcal{H}_1| \leq 2 \cdot \eta + k$  follows from Lemma 8, Lemma 15 and Lemma 16. The statement  $|\mathcal{H}_{-1}^1| \leq 2 \cdot \eta + 2k$  then follows from Lemma 13.  $\blacktriangleleft$

Next, we analyze  $|\mathcal{H}_0^1|$  with the notion of troublemakers defined in Section 3.1. In particular, for two rounds  $t, t'$  with  $t' < t$ , we say  $t'$  is the *parent* of  $t$  and  $t$  is the *child* of  $t'$  if  $t \in \mathcal{H}_0^1$  and  $t' = \max\{t'' < t \mid \sigma_t \in \hat{C}_{t''}\}$ .

► **Lemma 18.** *Every round  $t \in \mathcal{H}_0^1 \setminus \{\tilde{t} \mid R_{\tilde{t}}(\sigma_{\tilde{t}}) = -1\}$  has one parent  $t' \in \Gamma \cup \mathcal{H}_1$ , and any round  $t'$  has at most one child.*



**Proof.** Since  $t \notin \{\tilde{t} \mid R_{\tilde{t}}(\sigma_{\tilde{t}}) = -1\}$ , the page  $\sigma_t$  is requested at round  $r = R_t(i) \in [1, t)$ , which gives that  $\sigma_t \in \hat{C}_{r+1}$ . As  $t \in \mathcal{H}_0^1$ , we know that  $\sigma_t \notin \hat{C}_t$ . Therefore, there must exist a round  $t'' \in [r+1, t)$  with  $\sigma_t = \hat{e}_{t''} \in \hat{C}_{t''}$ . Since  $\{t'' \mid t'' < t \wedge \sigma_t \in \hat{C}_{t''}\} \neq \emptyset$ , the existence of the parent round of  $t$  is ensured.

For the parent  $t'$  of round  $t$ , we know that  $\hat{e}_{t'} = \sigma_t$ , because  $\sigma_t \in \hat{C}_{t'} \setminus \hat{C}_{t'+1}$ . Then we have  $\hat{e}_{t'} \neq \perp$  and  $\hat{e}_{t'} \in I_{t'}$ , where the second equality holds because  $\sigma_t \in C_t$  and  $\sigma_{t''} \neq \sigma_t$  for every  $t'' \in [t', t-1]$ . Then by the definitions of the parent round and  $\mathcal{H}_0^1$ , we have  $A_{t'}(\hat{e}_{t'}) = t \in [T]$  and  $\hat{e}_{t'} \in C_t$ , which means that  $\hat{e}_{t'} \in C_{A_{t'}(\hat{e}_{t'})}$ . Therefore,  $t' \in \Gamma$  if  $e_{t'} = \perp$  or  $e_{t'} \notin I_{t'}$ . If  $e_{t'} \neq \perp$  and  $e_{t'} \in I_{t'}$ , then we have  $e_{t'} \neq \hat{e}_{t'}$ , because otherwise  $\hat{e}_{t'} \notin C_t$ . By Lemma 12, in such a case we have  $t' \in \mathcal{H}_1$ .

It remains to prove that any round  $t'$  has at most one child. Suppose that  $t'$  has two children  $t_1, t_2$  with  $t_1 < t_2$ . In such a case,  $\sigma_{t_2} = \hat{e}_{t'} = \sigma_{t_1} \in \hat{C}_{t_1+1}$ , which conflicts with the definition of the parent round.  $\blacktriangleleft$

Putting Lemma 9, Lemma 17, and Lemma 18 together gives the following result.

► **Theorem 19.**  $\text{cost}(\text{Sim}) - \text{OPT} \leq 4\eta + 4k + |\Gamma| - |\mathcal{H}^{-1}|$ .

We defer the proof of Theorem 19 to the full version [11]. The upper bound on  $|\Gamma| - |\mathcal{H}^{-1}|$  is studied in the next subsection.

### 3.3 Labeling for Troublemakers

From the high level, the analysis in this part on  $|\Gamma| - |\mathcal{H}^{-1}|$  is done by showing that each troublemaker  $\gamma$  either can be mapped a distinct round in  $\mathcal{H}^{-1}$ , or can be mapped to a round  $t < \gamma$  that has a prediction error. We specify the mappings with a procedure called **Labeling**, which is designed to avoid mapping too many troublemakers to a single prediction error. Notice that **Labeling** is only used in the analysis, while the paging algorithm is unaware of the output generated by **Labeling**.

Procedure **Labeling** takes  $\langle \{A_t(i)\}_{t \in [T], i \in [n]}, \{a_t\}_{t \in [T]} \rangle$  as the input, which implicitly encodes the operations of **FitF** and **Sim**, and for each troublemaker  $\gamma \in \Gamma$ , **Labeling** outputs a labeling function  $\lambda_\gamma : \theta_\gamma \mapsto ([n] \cup \perp)$ , which maps each round in the active period  $\theta_\gamma$  of  $\gamma$  to either a page  $i$  or an empty value. For each  $\gamma \in \Gamma$  and each round  $t$  in the active period  $\theta_\gamma$  with  $\lambda_\gamma(t) \neq \perp$ , we say that page  $i = \lambda_\gamma(t)$  is labelled by  $\gamma$ . Procedure **Labeling** is presented in Algorithm 1 with notions defined as follows.

$$\forall t \in [T] : \quad \mathcal{L}_t \doteq \bigcup_{\gamma \in \Gamma_t} \lambda_\gamma(t), \quad \text{and} \quad \Phi_t \doteq \hat{C}_t \setminus (C_t \cup \mathcal{L}_t),$$

$$\forall t \in [2, T] : \quad \Psi_t \doteq \hat{C}_t \setminus (C_t \cup \mathcal{L}_{t-1}).$$

Briefly speaking, for every  $\gamma \in \Gamma$ , **Labeling** picks an arbitrary page  $i = \hat{\delta}_\gamma$  from  $\Phi_\gamma$  and labels  $i$  with  $\gamma$  for the first round in the active period of  $\gamma$ , which means setting  $\lambda_\gamma(\gamma+1) = i$ . For convenience, the NAT of  $i$  after  $\gamma$  is denoted by  $\tau_\gamma$ . Then we consider the following cases.

- $\tau_\gamma > A_\gamma(\hat{e}_\gamma)$ : Then the label on  $\hat{\delta}_\gamma$  is kept throughout the active period  $\theta_\gamma$  of  $\gamma$ .
- $\tau_\gamma < A_\gamma(\hat{e}_\gamma)$  and  $\hat{\delta}_\gamma \in \hat{C}_{\tau_\gamma}$ : This means that  $\hat{\delta}_\gamma$  is not evicted by **Sim** before its NAT after  $\gamma$ . In such a case, the label on  $\hat{\delta}_\gamma$  is kept until the last round before its NAT.
- $\tau_\gamma < A_\gamma(\hat{e}_\gamma)$  and  $\hat{\delta}_\gamma \notin \hat{C}_{\tau_\gamma}$ : In such a case, a labelled page is evicted by **Sim** before its NAT after  $\gamma$ . For each round  $t$  with such an eviction, we label a new page at round  $t+1$  in  $\Psi_{t+1}$  with  $\gamma$ . We stop labelling new pages either when the labelled page is requested, or the NAT of the previous labelled page after the previous round is less than the NAT of the current labelled page.

---

**Algorithm 1** Procedure Labeling.
 

---

**Input:**  $\{A_t(i)\}_{t \in [T], i \in [n]}, \{a_t\}_{t \in [T]}$   
**Output:**  $\{\lambda_\gamma\}_{\gamma \in \Gamma}$

```

1 for each  $\gamma \in \Gamma$  do
2   Pick an arbitrary element  $i \in \Phi_\gamma$  and set  $\hat{\delta}_\gamma = i$ ;
3   Set  $\lambda_\gamma(\gamma + 1) = \hat{\delta}_\gamma$  and  $\tau_\gamma = A_\gamma(\hat{\delta}_\gamma)$ ;
4   if  $\tau_\gamma > A_\gamma(\hat{e}_\gamma)$  then
5     Set  $\lambda_\gamma(t) = \hat{\delta}_\gamma$  for all the remaining rounds  $t$  in  $\theta_\gamma$ ;
6   else
7     Set  $t = \gamma + 2$ ;
8     while  $t < A_\gamma(\hat{e}_\gamma)$  do
9       if  $\lambda_\gamma(t - 1) = \sigma_{t-1}$  then
10        break;
11       if  $\lambda_\gamma(t - 1) \neq \hat{e}_{t-1}$  then
12        Set  $\lambda_\gamma(t) = \lambda_\gamma(t - 1)$ ;
13       else
14        Pick an arbitrary element  $i$  from  $\Psi_t$  and set  $\lambda_\gamma(t) = i$ ;
15        if  $A_{t-1}(\lambda_\gamma(t)) > A_{t-1}(\lambda_\gamma(t - 1))$  then
16          Set  $\lambda_\gamma(t') = i$  for every round  $t' \in [t + 1, \min\{A_\gamma(\hat{e}_\gamma), A_t(i)\}]$ ;
17          Set  $t = \min\{A_\gamma(\hat{e}_\gamma), A_t(i)\} - 1$ ;
18          break;
19        Set  $t = t + 1$ ;
20    Set  $\lambda_\gamma(t') = \perp$  for every  $t' \in [t, A_\gamma(\hat{e}_\gamma)]$ ;
  
```

---

Before describing how Procedure **Labeling** is applied to map each troublemaker to a round with a prediction error or a round in  $\mathcal{H}^{-1}$ , we first prove that this procedure is consistent by showing that  $\Phi_t$  (resp.  $\Psi_t$ ) is not empty whenever we need to find a new page to label from  $\Phi_t$  (resp.  $\Psi_t$ ). For every troublemaker round  $t \in \Gamma$ , define

$$\zeta_t = \begin{cases} \sigma_t & \text{if } e_t = \perp \\ e_t & \text{otherwise} \end{cases}.$$

Then we have the following results.

► **Lemma 20.** *For each troublemaker round  $t \in \Gamma$ , we have (1)  $\zeta_t \neq \perp$ , (2)  $\zeta_t \in C_t \setminus \hat{C}_t$ , and (3) for any  $\gamma \in \Gamma_t$ , we have  $\zeta_t \neq \hat{e}_\gamma$ .*

**Proof.** Claim (1) and (2) are obvious. Now consider claim (3). Since  $\gamma < t < A_\gamma(\hat{e}_\gamma)$ ,  $\sigma_t \neq \hat{e}_\gamma$ , because otherwise  $t = A_\gamma(\hat{e}_\gamma)$ . By the definition of troublemakers,  $\hat{e}_\gamma \in C_{A_\gamma(\hat{e}_\gamma)}$ , so for any  $t \in [\gamma + 1, A_\gamma(\hat{e}_\gamma) - 1]$ , it holds that  $e_t \neq \hat{e}_\gamma$ . ◀

► **Lemma 21.** *For each troublemaker round  $t \in \Gamma$ , it holds that  $|\Phi_t| \geq 1$ .*

**Proof.** By Lemma 10 and Lemma 20, we have  $C_t \setminus \hat{C}_t \supseteq \{\hat{e}_\gamma\}_{\gamma \in \Gamma_t} \cup \{\zeta_t\}$ . Still by Lemma 20, it follows that  $\zeta_t \neq \hat{e}_\gamma$  for every  $\gamma \in \Gamma_t$ , then  $|C_t \setminus \hat{C}_t| \geq |\{\hat{e}_\gamma\}_{\gamma \in \Gamma_t}| + 1 = |\Gamma_t| + 1$ . Because  $|C_t| = |\hat{C}_t|$ , we have  $|\hat{C}_t \setminus C_t| \geq |\Gamma_t| + 1$ . Since for every  $\gamma$  with  $t \in \theta_\gamma$ , we have  $\gamma \in \Gamma_t$ , then it always holds that  $|\mathcal{L}_t| \leq |\Gamma_t|$ . This finishes the proof. ◀

► **Lemma 22.** *For each round  $t \in [T]$ , we have  $C_t \cap \mathcal{L}_t = \emptyset$ .*

**Proof.** This proposition is true because for any round  $t$  when we find a new page  $i$  to label,  $i \notin C_t$ , and the label on any page  $i$  is cancelled before  $i$  is requested.  $\blacktriangleleft$

► **Lemma 23.** *For every  $\gamma \in \Gamma$  and every  $t \in [\gamma + 2, A_\gamma(\hat{e}_\gamma) - 1]$ , if  $\hat{e}_{t-1} = \lambda_\gamma(t - 1)$ , then  $\Psi_t \neq \emptyset$ .*

**Proof.** First, consider the case where  $\Phi_{t-1} \neq \emptyset$ . For each page  $i \in \Phi_{t-1}$ , we know  $i \neq \hat{e}_{t-1}$  because  $i \notin \mathcal{L}_{t-1}$  while  $\hat{e}_{t-1} = \lambda_\gamma(t - 1) \in \mathcal{L}_{t-1}$ . This gives  $i \in \hat{C}_t$ . Moreover, we have  $i \neq \sigma_{t-1}$  because  $\hat{e}_{t-1} \neq \perp$ . This gives  $i \notin C_t$ . Therefore,  $i \in \hat{C}_t \setminus (C_t \cup \mathcal{L}_{t-1}) = \Psi_t$ .

Now consider  $\Phi_{t-1}$  is empty. In such a case,  $|\hat{C}_{t-1} \setminus C_{t-1}| = |\mathcal{L}_{t-1}|$ . Lemma 10 indicates that for every  $\gamma' \in \Gamma_{t'}$  with  $t' \in \theta_{\gamma'}$ , we have  $\hat{e}_{\gamma'} \in C_{t'} - \hat{C}_{t'}$ . Then  $|\hat{C}_{t-1} \setminus C_{t-1}| = |\mathcal{L}_{t-1}|$  implies that  $C_{t-1} \setminus \hat{C}_{t-1} = \{\hat{e}_{\gamma'}\}_{\gamma' \in \Gamma_{t-1}}$ . By the definition of active troublemakers,  $\sigma_{t-1} \notin C_{t-1} - \hat{C}_{t-1}$ . Also, we have  $\sigma_{t-1} \notin I_{t-1} = C_{t-1} \cap \hat{C}_{t-1}$ , because  $\hat{e}_{t-1} \neq \perp$ . Therefore,  $\sigma_{t-1} \notin C_{t-1}$ , which means that  $e_{t-1} \neq \perp$ . Still by the definition of the troublemakers, we have  $e_{t-1} \notin \{\hat{e}_{\gamma'}\}_{\gamma' \in \Gamma_{t-1}} = C_{t-1} \setminus \hat{C}_{t-1}$ . Thus,  $e_{t-1} \in I_{t-1} \subseteq \hat{C}_{t-1}$ . By Lemma 22, we have  $e_{t-1} \notin \mathcal{L}_{t-1}$  and  $e_{t-1} \neq \hat{e}_{t-1}$ . Putting  $e_{t-1} \neq \hat{e}_{t-1}$  and  $e_{t-1} \in \hat{C}_{t-1}$  together, we get  $e_{t-1} \in \hat{C}_t - C_t$ . Therefore,  $e_{t-1} \in \Psi_t$ .  $\blacktriangleleft$

Lemma 21 and Lemma 23 ensure the consistency of Procedure **Labeling**.

► **Lemma 24.** *For every round  $t$  and any page  $i \in [n]$ , there exists at most one troublemaker  $\gamma \in \Gamma_t$  so that  $\lambda_\gamma = i$ .*

The proof of Lemma 24 is deferred to the full version [11]. It also gives the following byproduct.

► **Lemma 25.** *For each round  $t \in [2, T]$ , we have  $|\mathcal{L}_t \setminus \mathcal{L}_{t-1}| \leq 1$ .*

Let  $t$  be an arbitrary round with  $\hat{e}_t \neq \perp$ . Suppose that there exists a page  $i \in \hat{C}_t$  satisfying  $A_t(i) > A_t(\hat{e}_t)$ ,  $i \notin \mathcal{L}_t$ , and  $i \in \mathcal{L}_{t'}$  for every  $t' \in [t + 1, \min\{A_t(\hat{e}_t), t^\circ\}]$ , where

$$t^\circ = \begin{cases} \min\{t'' \mid t'' > t \wedge \hat{e}_{t''} = i\} & \text{if } \exists t'' \in (t + 1, T] \text{ s.t. } \hat{e}_{t''} = i \\ \infty & \text{otherwise} \end{cases}. \quad (6)$$

Lemma 25 implies that such a page  $i$  is unique if it exists. In such a case, we say that the page  $i$  is the *competitor* of  $\hat{e}_t$ , and the round  $t$  has an *abettor* round  $t^*$  specified as follows. Let  $r = R_t(\hat{e}_t)$  and  $r' = R_t(i)$ , then the abettor of  $t$  is

$$t^* = \begin{cases} r & \text{if } r \neq -1 \wedge A_r(\sigma_r) \neq a_r \\ r' & \text{if } (r = -1 \vee A_r(\sigma_r) = a_r) \wedge (r' \neq -1 \wedge A_{r'}(\sigma_{r'}) \neq a_{r'}) \\ -1 & \text{otherwise} \end{cases}. \quad (7)$$

► **Lemma 26.** *For any round  $t^* \in [T]$ , the number of rounds in  $\{t \mid \hat{e}_t \neq \perp \wedge R_t(\hat{e}_t) \neq -1 \wedge t^* \text{ is the abettor of } t\}$  is at most two.*

**Proof.** It can be proved in a similar way with Lemma 16 that  $\{t \mid \hat{e}_t \neq \perp \wedge R_t(\hat{e}_t) \neq -1 \wedge t^* \text{ is the abettor of } t \wedge t^* = R_t(\hat{e}_t)\}$  contains at most one round. It remains to prove that  $\{t \mid \hat{e}_t \neq \perp \wedge R_t(\hat{e}_t) \neq -1 \wedge t^* \text{ is the abettor of } t \wedge t^* = R_t(i)\}$  contains at most a single round, where  $i$  is the competitor of  $\hat{e}_t$  as defined in Eq. (6). Notice that different from the case considered in the proof of Lemma 16, the page  $i$  may not be evicted by **FitF**. Therefore, we need to utilize the properties of procedure **Labeling** to prove the uniqueness of the round  $t$  which satisfies  $t^* = R_t(i)$ .

▷ **Claim 27.** If a round  $t$  has an abettor  $t^* = R_t(i)$  with  $i$  being the competitor of  $\hat{e}_t$  and  $r = R_t(\hat{e}_t) \neq -1$ , then it holds that  $\hat{a}_t(i) < Z$  and  $\hat{a}_{t'}(i) = Z$  for any round  $t' \in [A_t(\hat{e}_t), A_t(i))$ .

*Proof.* Since  $t^* = R_t(i)$  and  $r \neq -1$ , we have  $A_r(\sigma_r) = a_r$ , which by Lemma 7 gives  $A_t(\hat{e}_t) = \hat{a}_t(\hat{e}_t) < Z$ . Following Lemma 14, we have  $\hat{a}_t(i) \leq A_t(\hat{e}_t)$  and  $\hat{a}_t(i) < Z$ . If there exists a round  $t'' \in (t, A_t(\hat{e}_t))$  so that  $\hat{a}_{t''}(i) = Z$ , then it follows from Eq. (3) that for any round  $t' \in [A_t(\hat{e}_t), A_t(i))$ , it holds that  $\hat{a}_{t'}(i) = Z$ , which means that this proposition holds. We proceed to prove that  $\hat{a}_{A_t(\hat{e}_t)}(i) = Z$  if  $\hat{a}_{t''}(i) \neq Z$  holds for every round  $t'' \in (t, A_t(\hat{e}_t))$ . In such a case, it can be inferred from Eq. (3) that  $\hat{a}_{t''}(i) = \hat{a}_t(i)$ . Since  $A_r(\sigma_r) = a_r$ , Lemma 7 indicates that  $\hat{a}_{t''}(\hat{e}_t) = A_{t''}(\hat{e}_t) = A_t(\hat{e}_t) = \hat{a}_t(\hat{e}_t)$  and  $\hat{a}_{t''}(\hat{e}_t) < Z$  hold for any round  $t'' \in (t, A_t(\hat{e}_t))$ . Combining  $\hat{a}_{t''}(\hat{e}_t) = \hat{a}_t(\hat{e}_t)$  with  $\hat{a}_{t''}(i) = \hat{a}_t(i)$  gives  $\hat{a}_{t''}(\hat{e}_t) \geq \hat{a}_{t''}(i)$ . Therefore,  $\hat{a}_{A_t(\hat{e}_t)-1}(i) \leq \hat{a}_{A_t(\hat{e}_t)-1}(\hat{e}_t) < Z$ . Moreover, we have  $\hat{a}_{A_t(\hat{e}_t)-1}(i) < A_t(\hat{e}_t)$  because  $A_t(\hat{e}_t) \geq \hat{a}_t(i) = \hat{a}_{t''}(i)$  holds for every  $t'' \in (t, A_t(\hat{e}_t))$ . By Eq. (3), the conditions for  $\hat{a}_{A_t(\hat{e}_t)}(i) = Z$  are all satisfied. Thus, the equality  $\hat{a}_{t'}(i) = Z$  holds for any round  $t' \in [A_t(\hat{e}_t), A_t(i))$ . ◀

For any round  $t' \in (t^*, t)$ , the round  $t^*$  cannot be the abettor of  $t'$  with taking  $t^* = R_{t'}(i)$ , because otherwise,

- if  $A_{t'}(\hat{e}_{t'}) \geq t$ , then by the definition of abettors, we have  $i \in \mathcal{L}_t$ , which conflicts with the requirement in the definition of abettors; else
- if  $A_{t'}(\hat{e}_{t'}) < t$ , then we get  $\hat{a}_t(i) = Z$  with using the second statement in Claim 27, which conflicts with the first statement in Claim 27.

Therefore, this proposition holds. ◀

The following result can be proved by following the same line of arguments with the proof of Lemma 15.

▶ **Lemma 28.** Let  $t$  be an arbitrary round that has an abettor  $t^*$ . If  $t^* = -1$ , then  $R_t(\hat{e}_t) = -1$ . If  $R_t(\hat{e}_t) \neq -1$ , then there exists a round  $t'$  such that  $\{t^*, t'\} \in \text{INV}$ .

▶ **Remark 29.** Notice that the statement of Lemma 28 is consistent because for any round  $t$  having an abettor, by definition we have  $\hat{e}_t \neq \perp$ .

For an arbitrary round  $t$ , if there exists a page  $i$  in  $\hat{C}_t$  satisfies (1)  $A_t(i) \leq T$ , (2)  $i \notin \mathcal{L}_t$  and (3)  $i \in \mathcal{L}_{t'}$  for every  $t' \in [t+1, A_t(i)]$ , we say that  $t^\Delta = A_t(i)$  is the *savior* of  $t$ .

▶ **Lemma 30.** If a round  $t$  has a savior  $t^\Delta$ , then (1)  $t^\Delta \in \mathcal{H}^{-1}$ , and (2) for any  $t^\Delta \in \mathcal{H}^{-1}$ , it is the savior of at most one step  $t$ .

**Proof.** The first claim follows from the definition of  $\mathcal{H}^{-1}$ . For any round  $t' \in [t+1, t^\Delta]$ ,  $t^\Delta$  is not the savior of  $t'$ , because  $\sigma_{t^\Delta} \in \mathcal{L}_{t'}$ . Therefore, the second claim holds. ◀

▶ **Theorem 31.**  $|\Gamma| - |\mathcal{H}^{-1}| \leq 2 \cdot \eta + k$ .

**Proof.** The main idea of this proof is to show that each troublemaker can be mapped to a distinct *broker* round, and each broker either has an abettor or has a savior. In particular, we classify the troublemakers  $\gamma \in \Gamma$  into the following three categories.

1.  $\{\gamma \in \Gamma \mid \tau_\gamma > A_\gamma(\hat{e}_\gamma)\}$ : Here, the troublemaker  $\gamma$  as a round has an abettor  $t^*$ , because  $\hat{\delta}_\gamma \in \mathcal{L}_t$  for every step  $t \in [\gamma+1, \min\{A_\gamma(\hat{e}_\gamma), t^\circ\}]$  where  $t^\circ$  is defined in the same way with Eq. (6). In such a case, we say  $\gamma$  is the broker of itself.
2.  $\{\gamma \in \Gamma \mid \tau_\gamma < A_\gamma(\hat{e}_\gamma) \wedge \hat{\delta}_\gamma \in \hat{C}_{\tau_\gamma}\}$ : Now the troublemaker  $\gamma$  as a round has a savior  $\tau_\gamma$ , because by the definition of the troublemaker,  $A_\gamma(\hat{e}_\gamma) \leq T$ , which gives  $\tau_\gamma \leq T$ . Moreover, it holds for every round  $t \in [\gamma+1, A_\gamma(\hat{\delta}_\gamma)]$  that  $\hat{\delta}_\gamma \in \mathcal{L}_t$ . The broker for such a troublemaker  $\gamma$  is also itself.

3.  $\{\gamma \in \Gamma \mid \tau_\gamma < A_\gamma(\hat{e}_\gamma) \wedge \hat{\delta}_\gamma \notin \hat{C}_{\tau_\gamma}\}$ : In such a case, let  $i$  be the last page labelled by  $\gamma$  and  $t$  be the first round with  $\lambda_\gamma(t) = i$ . Since  $\hat{\delta}_\gamma \notin \hat{C}_{\tau_\gamma}$ , we have  $t-1 \in \theta_\gamma$ . Let  $i' = \lambda_\gamma(t-1)$ , then we have  $i' \in \hat{C}_{t-1}$  because  $\hat{e}_{t-1} = i'$ . Also, we have  $i \in \hat{C}_{t-1}$ , because  $i$  is chosen from  $\Psi_t \subseteq \hat{C}_t \setminus C_t$  and  $i \neq \hat{e}_{t-1}$ . Now consider two subcases.
- $A_{t-1}(i') < A_{t-1}(i)$ : In such a case, the round  $t-1$  has an abettor  $t^*$ , because we have  $i \in \mathcal{L}_{t^*}$  for every  $t^* \in [t, \min\{A_{t-1}(i), \tau_\gamma, t^\circ\}]$ , which satisfies the requirement in the definition of abettors because  $\tau_\gamma > A_{t-1}(i')$ .
  - $A_{t-1}(i') > A_{t-1}(i)$ : It can be proved inductively that  $A_{t-1}(i) < \tau_\gamma$ , which implies that  $A_{t-1}(i) \leq T$ . By definition, it follows that  $t-1$  has a savior  $t^\Delta = A_{t-1}(i)$ .
- The round  $t-1$  is said to be the *broker* of the troublemaker  $\gamma$ . Lemma 24 ensures that the round  $t-1$  cannot be the broker of two different troublemakers, because  $\hat{e}_{t-1} = \lambda_\gamma(t-1)$ . Moreover, by Lemma 25, the broker  $t-1$  is not a troublemaker.

To sum up, each troublemaker  $\gamma$  can be mapped to a distinct broker  $t$ , and each broker  $t$  either has an abettor or has a savior. Then by Lemma 26, Lemma 28 and Lemma 30, this theorem holds.  $\blacktriangleleft$

The following result is obtained by combining Theorem 19 and Theorem 31.

► **Theorem 32.** *It follows that  $\text{cost}(\text{Sim}) - \text{OPT} \leq 6\eta + 5k$ .*

Because the cumulative prediction error is assumed to satisfy  $\eta \in o(T)$ , we have  $\text{cost}(\text{Sim}) - \text{OPT} \in o(T)$ . Therefore, **Sim** has the vanishing regret when there is a single predictor.

## 4 Multiple NAT Predictors

This section extends the result obtained in Section 3 to the general case where there are  $M > 1$  predictors making NAT predictions under the bandit access model. Our results on the full information access model are deferred to the full version [11].

For the bandit access model, in this part, we design an algorithm called *Sightless Chasing and Switching* (**S-C&S**) and prove that it has the vanishing regret.

The procedure of **S-C&S** is described in Algorithm 2. It is assumed that **S-C&S** is provided with blackbox accesses to the online algorithm *Implicitly Normalized Forecaster* (**INF**) [2] for the *Multiarmed Bandit Problem* (**MBP**) [3]. The MBP problem is an online problem defined over  $\Upsilon \in \mathbb{Z}_{>0}$  rounds and a set  $X$  of arms. An oblivious adversary specifies a cost function  $F_v : X \mapsto [0, 1]$  for each round  $v \in [\Upsilon]$  that maps each arm  $x \in X$  to a cost in  $[0, 1]$ . An algorithm for MBP needs to choose an arm  $x_v$  at the beginning of each round  $v \in [\Upsilon]$ , and then the cost  $F_v(x_v)$  incurred by the chosen arm  $x_v$  is revealed to the algorithm. The objective of MBP is to minimize the cumulative cost incurred by the chosen arms  $\{x_v\}_{v \in [\Upsilon]}$ .

► **Theorem 33** ([2]). *The algorithm **INF** ensures that the chosen arms  $\{x_v\}_{v \in [\Upsilon]}$  satisfy*

$$\sum_{v \in [\Upsilon]} F_v(x_v) - \min_{x^* \in X} \sum_{v \in [\Upsilon]} F_v(x^*) \in O(\sqrt{|X| \cdot \Upsilon}).$$

Our algorithm **S-C&S** partitions the rounds into consecutive epochs of length  $\tau \in \mathbb{Z}_{>0}$  and initializes **INF** by setting  $\Upsilon = \lceil \frac{T}{\tau} \rceil$  and  $X = [M]$ , which means that each epoch in the online paging problem is mapped to a round in MBP, and each predictor is taken as an arm. The choice of the value for  $\tau$  is discussed later. At the beginning of the first round  $t_1^v = (v-1)\tau + 1$  in each epoch  $v \in [\Upsilon]$ , **S-C&S** accesses **INF** to pick one predictor  $j_{t_1^v}$ . Then,

---

**Algorithm 2** Algorithm S-C&S.

---

**Input:**  $\{\sigma_t\}_{t \in [T]}$ , MBP algorithm INF, initial cache profile  $\hat{C}_1$   
**Output:**  $\{\hat{e}_t\}_{t \in [T]}$

- 1 Initialize the MBP algorithm INF with the number of rounds  $\Upsilon = \lceil T/\tau \rceil$  and the set of arms  $X = [M]$ ;
- 2 **for** each round  $t \in [T]$  **do**
- 3   **if**  $t \bmod \tau = 1$  **then**
- 4     Invoke INF to choose a predictor  $j_t \in [M]$ ;
- 5     Set  $j = j_t$ ;
- 6   **else**
- 7     Set  $j = j_{t'}$  with  $t' = \lfloor t/\tau \rfloor \cdot \tau + 1$ ;
- 8   Query the predictor  $j$  to obtain the prediction  $a_t^j$ ;
- 9   **for** each page  $i \in [n]$  **do**
- 10     **if**  $i = \sigma_t$  **then**
- 11       Set  $\hat{a}_t(i) = a_t^j$ ;
- 12     **else if**  $t \bmod \tau = 1$  **then**
- 13       Set  $\hat{a}_t(i) = Z + 1$ ;
- 14     **else if**  $\hat{a}_{t-1}(i) = t \wedge i \neq \sigma_t \wedge \hat{a}_{t-1}(i) \leq \hat{a}_{t-1}(\sigma_t) < Z$  **then**
- 15       Set  $\hat{a}_t(i) = Z$ ;
- 16     **else**
- 17       Set  $\hat{a}_t(i) = \hat{a}_{t-1}(i)$ ;
- 18   **if**  $\sigma_t \notin \hat{C}_t$  **then**
- 19     Set  $\hat{e}_t$  be the page  $i \in C_t$  that maximizes  $\hat{a}_t(i)$  with breaking ties arbitrarily;
- 20     Update  $\hat{C}_{t+1} = (\hat{C}_t \setminus \{\hat{e}_t\}) \cup \{\sigma_t\}$ ;
- 21   **else**
- 22     Set  $\hat{e}_t = \perp$ , and set  $\hat{C}_{t+1} = \hat{C}_t$ ;
- 23   **if**  $t \bmod \tau = 0$  **then**
- 24     Set  $f = 0$ ;
- 25     **for** each round  $t' \in [t - \tau + 1, t]$  **do**
- 26       **if**  $(t' = t - \tau + 1) \vee (\hat{e}_{t'} \neq \perp) \vee ((\hat{e}_{t'} = \perp) \wedge (t' > t - \tau + 1) \wedge (\hat{a}_{t'-1}(\sigma_{t'}) = Z + 1))$  **then**
- 27         Set  $f = f + 1$ ;
- 28     Send  $\frac{f}{\tau}$  to INF as the cost incurred by  $j_{t-\tau+1}$  in the epoch  $\frac{t}{\tau}$ ;

---

S-C&S simulates the algorithm **Sim**, which is proposed in Section 3, throughout the epoch  $v$  with taking  $t_1^v$  as its initial round,  $\hat{C}_{t_1^v}$  as its initial cache profile, and  $j_{t_1^v}$  as the single predictor. At the end of the last round  $t_\tau^v = v \cdot \tau$  in epoch  $v$ , S-C&S sends

$$F_v(j_{t_1^v}) = \frac{1}{n} \left| \left\{ t' \in [t_1^v, t_\tau^v] \mid (t' = t_1^v) \vee (\hat{e}_{t'} \neq \perp) \vee ((\hat{e}_{t'} = \perp) \wedge (t' > t_1^v) \wedge (\hat{a}_{t'-1}(\sigma_{t'}) = Z + 1)) \right\} \right| \quad (8)$$

to INF as the cost  $F_v(j_{t_1^v})$  of choosing  $j_{t_1^v}$  for  $v$ .

Notice that in MBP, the cost functions are generated by an oblivious adversary. We take this setting as a requirement that the cost function  $F_v$  for each round  $v$  in MBP should not depend on the arms chosen in the previous rounds  $x_1, x_2, \dots, x_{v-1}$ . The following result shows that by feeding INF a cost that can be larger than the normalized cost that is actually incurred by S-C&S in the epoch, this requirement is satisfied.

► **Lemma 34.** Let  $F_v(j_{t_1^v} | j_{t_1^1}, \dots, j_{t_1^{v-1}})$  be the cost sent by  $\mathcal{S}\text{-}\mathcal{C}\&\mathcal{S}$  to  $\text{INF}$  at the end of an arbitrary epoch  $v$  conditioned on the predictors chosen for the previous epochs  $j_{t_1^1}, \dots, j_{t_1^{v-1}}$ . Then for any different sequence of predictors  $\tilde{j}_{t_1^1}, \dots, \tilde{j}_{t_1^{v-1}}$ , we have  $F_v(j_{t_1^v} | j_{t_1^1}, \dots, j_{t_1^{v-1}}) = F_v(j_{t_1^v} | \tilde{j}_{t_1^1}, \dots, \tilde{j}_{t_1^{v-1}})$ .

**Proof.** For the epoch  $v$ , a round  $t$  is said to be *fresh* if for any earlier round  $t' < t$  in  $v$ , it holds that  $\sigma_t \neq \sigma_{t'}$ . We first consider the case where there are at least  $k$  fresh rounds in the epoch, which means that at least  $k$  different pages are requested. Let  $t$  be the first round after the first  $k$  fresh rounds. We first consider the interval  $[t_1^v, t - 1]$ . For each page  $i \in [n]$  and each round  $t' \in [t_1^v, t - 1]$ , we say  $i$  is *marked* at  $t'$  if there exists a round  $t'' \in [t_1^v, t']$  with  $\sigma_{t''} = i$ , otherwise  $i$  is said to be *unmarked* at  $t'$ . Then it can be inferred from Lemma 7 that  $\hat{a}_{t'}(i) = Z + 1$  if  $i$  is unmarked at  $t'$ , and  $\hat{a}_{t'}(i) \leq Z$  if  $i$  is marked at  $t'$ . Thus, there is no marked page getting evicted before  $t$ , and any round  $t' \in [t_1^v, t - 1]$  that satisfies  $\hat{e}_{t'} \neq \perp$  must be a fresh round.

► **Claim 35.** For each round  $t' \in [t_1^v, t - 1]$ , it is counted by Eq. (8) if and only if  $t'$  is fresh.

**Proof.** By definition, the round  $t_1^v$  is a fresh round. Now consider a round  $t' \in [t_1^v + 1, t - 1]$ . As mentioned above, if  $\hat{e}_{t'} \neq \perp$ , then  $t'$  is fresh. Also, if  $t'$  is fresh, then it can be inferred from Lemma 7 that  $\hat{a}_{t'-1}(\sigma_{t'}) = Z + 1$ . If  $t'$  is not fresh, which means that  $\sigma_{t'}$  is marked, then it holds that  $\hat{e}_{t'} = \perp$  and  $\hat{a}_{t'-1}(\sigma_{t'}) \leq Z$ . Therefore, this claim holds. ◀

Therefore, the contribution of the rounds in  $[t_1^v, t - 1]$  to  $F_v(j_{t_1^v})$  is always  $\frac{k}{n}$ , which is independent of  $j_{t_1^1}, \dots, j_{t_1^{v-1}}$ . A similar result can also be obtained when there are less than  $k$  fresh rounds in the epoch  $v$ .

At the beginning of round  $t$ ,  $\hat{C}_t$  contains exactly the first  $k$  different pages required in the epoch  $v$ , and for each page  $i \in \hat{C}_t$ , the remedy prediction  $\hat{a}_t(i)$  is computed only based on  $\{a_{t'}^{j_{t_1^v}}\}_{t' \in [t_1^v, t]}$  and  $\{\sigma_{t'}\}_{t' \in [t_1^v, t]}$ . Therefore, for any  $t' \geq t$ , the decision on  $\hat{e}_{t'}$  is independent of the choices over  $j_{t_1^1}, \dots, j_{t_1^{v-1}}$ .

Furthermore, since at round  $t - 1$ , every page  $i \in \hat{C}_{t-1}$  is marked, then for any round  $t' \geq t$  with  $\hat{e}_t = \perp$ , the page  $\sigma_{t'}$  has been requested at least once in the interval  $[t_1^v, t' - 1]$ , which means that  $\hat{a}_{t'-1}(\sigma_{t'}) \leq Z$ . This observation is formally stated in the following claim.

► **Claim 36.** For any round  $t' \geq t$ , it is counted by Eq. (8) if and only if  $\hat{e}_{t'} \neq \perp$ .

Thus, the contribution of the rounds in  $[t, v \cdot \tau]$  to  $F_v(j_{t_1^v})$  does not depend on the previous epochs  $\{j_{t_1^1}, \dots, j_{t_1^{v-1}}\}$ , either. ◀

For an epoch  $v \in \Upsilon$  and a predictor  $j \in [M]$  chosen for  $v$ , let  $\sharp\text{Evictions}_v(j) = |\{t \in [t_1^v, t_\tau^v] \mid \hat{e}_t \neq \perp\}|$ . The following result is also obtained by combining Claim 35 with Claim 36.

► **Lemma 37.** For each epoch  $v$ ,  $\sharp\text{Evictions}_v(j) \leq \tau \cdot F_v(j) \leq \sharp\text{Evictions}_v(j) + k$ .

The following result is obtained by putting Theorem 32, Theorem 33, Lemma 34, and Lemma 37 together.

► **Theorem 38.** By taking  $\tau = \lfloor T^{\frac{1}{3}} \rfloor$ , the regret of  $\mathcal{S}\text{-}\mathcal{C}\&\mathcal{S}$  is bounded by  $O(kT^{\frac{2}{3}}\sqrt{M} + \eta^{\min})$ .

The proof of this theorem is deferred to the full version [11]. By assumption, we have  $\eta^{\min} \in o(T)$ . Therefore, Theorem 38 implies that  $\mathcal{S}\text{-}\mathcal{C}\&\mathcal{S}$  has a vanishing regret.

Note that the result in this section cannot be obtained by using the results in [12] directly, because the algorithms proposed in [12] requires to know the cache profile of the algorithm  $\text{Sim}$  that follows each predictor, which is unavailable under the bandit access model.



## References

- 1 Antonios Antoniadis, Christian Coester, Marek Eliás, Adam Polak, and Bertrand Simon. Online metric algorithms with untrusted predictions. In *Proceedings of Machine Learning and Systems*, ICML '20, pages 11463–11473. PMLR, 2020. URL: [https://proceedings.icml.cc/static/paper\\_files/icml/2020/6657-Paper.pdf](https://proceedings.icml.cc/static/paper_files/icml/2020/6657-Paper.pdf).
- 2 Jean-Yves Audibert and Sébastien Bubeck. Minimax policies for adversarial and stochastic bandits. In *COLT '09 - The 22nd Conference on Learning Theory*, Montreal, Quebec, Canada, June 2009. URL: <http://www.cs.mcgill.ca/%7Ecolt2009/papers/022.pdf#page=1>.
- 3 Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing*, 32(1):48–77, 2002. doi:10.1137/S0097539701398375.
- 4 Grant Ayers, Heiner Litz, Christos Kozyrakis, and Parthasarathy Ranganathan. Classifying memory access patterns for prefetching. In *Architectural Support for Programming Languages and Operating Systems*, ASPLOS '20, pages 513–526, Lausanne, Switzerland, March 2020. ACM. doi:10.1145/3373376.3378498.
- 5 Laszlo A. Belady. A study of replacement algorithms for virtual-storage computer. *IBM Systems Journal*, 5(2):78–101, 1966. doi:10.1147/sj.52.0078.
- 6 Avrim Blum and Carl Burch. On-line learning and the metrical task system problem. *Machine Learning*, 39(1):35–58, 2000. doi:10.1023/A:1007621832648.
- 7 Allan Borodin and Ran El-Yaniv. *Online computation and competitive analysis*. Cambridge University Press, 1998.
- 8 Allan Borodin, Nathan Linial, and Michael E. Saks. An optimal on-line algorithm for metrical task system. *J. ACM*, 39(4):745–763, 1992. doi:10.1145/146585.146588.
- 9 Peter Braun and Heiner Litz. Understanding memory access patterns for prefetching. In *International Workshop on AI-assisted Design for Architecture (AIDArc), held in conjunction with ISCA*, 2019. URL: <https://eecs.oregonstate.edu/aidarc/paper/MAP.pdf>.
- 10 Nicolo Cesa-Bianchi and Gabor Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, USA, 2006. doi:10.1017/CB09780511546921.
- 11 Yuval Emek, Shay Kutten, and Yangguang Shi. Online paging with a vanishing regret. *arXiv preprint*, 2020. arXiv:2011.09439.
- 12 Yuval Emek, Ron Lavi, Rad Niazadeh, and Yangguang Shi. Stateful posted pricing with vanishing regret via dynamic deterministic markov decision processes. *Advances in Neural Information Processing Systems (NeurIPS '20)*, 33, 2020. URL: <https://papers.nips.cc/paper/2020/file/1f10c3650a3aa5912dccc5789fd515e8-Paper.pdf>.
- 13 Amos Fiat, Richard M. Karp, Michael Luby, Lyle A. McGeoch, Daniel Dominic Sleator, and Neal E. Young. Competitive paging algorithms. *Journal of Algorithms*, 12(4):685–699, 1991. doi:10.1016/0196-6774(91)90041-V.
- 14 Sreenivas Gollapudi and Debmalya Panigrahi. Online algorithms for rent-or-buy with expert advice. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *ICML '19*, pages 2319–2327, Long Beach, California, USA, June 2019. PMLR. URL: <http://proceedings.mlr.press/v97/gollapudi19a.html>.
- 15 Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *3rd International Conference on Learning Representations*, ICLR '15, San Diego, CA, USA, May 2015. arXiv:1412.6572.
- 16 Milad Hashemi, Kevin Swersky, Jamie A. Smith, Grant Ayers, Heiner Litz, Jichuan Chang, Christos Kozyrakis, and Parthasarathy Ranganathan. Learning memory access patterns. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *ICML '18*, pages 1924–1933, Stockholmsmässan, Stockholm, Sweden, July 2018. PMLR. URL: <http://proceedings.mlr.press/v80/hashemi18a.html>.
- 17 Evan Zheran Liu, Milad Hashemi, Kevin Swersky, Parthasarathy Ranganathan, and Junwhan Ahn. An imitation learning approach for cache replacement. In *Proceedings of the 37th International Conference on Machine Learning*, ICML '20, July 2020. arXiv:2006.16239.

- 18 Thodoris Lykouris and Sergei Vassilvitskii. Competitive caching with machine learned advice. In *Proceedings of the 35th International Conference on Machine Learning, ICML '18*, volume 80, pages 3302–3311, Stockholmsmässan, Stockholm, Sweden, July 2018. PMLR. URL: <http://proceedings.mlr.press/v80/lykouris18a.html>.
- 19 Leeor Peled, Uri C. Weiser, and Yoav Etsion. A neural network prefetcher for arbitrary memory access patterns. *ACM Transactions on Architecture and Code Optimization (TACO)*, 16(4):37:1–37:27, 2020. doi:10.1145/3345000.
- 20 Dhruv Rohatgi. Near-optimal bounds for online caching with machine learned advice. In *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA '20*, pages 1834–1845, Salt Lake City, UT, USA, January 2020. SIAM. doi:10.1137/1.9781611975994.112.
- 21 Daniel D. Sleator and Robert E. Tarjan. Amortized efficiency of list update and paging rules. *Commun. ACM*, 28(2):202–208, 1985. doi:10.1145/2786.2793.
- 22 Ajitesh Srivastava, Angelos Lazaris, Benjamin Brooks, Rajgopal Kannan, and Viktor K. Prasanna. Predicting memory accesses: the road to compact ml-driven prefetcher. In *Proceedings of the International Symposium on Memory Systems, MEMSYS '19*, pages 461–470, Washington, DC, USA, September – October 2019. ACM. doi:10.1145/3357526.3357549.
- 23 Ajitesh Srivastava, Ta-Yang Wang, Pengmiao Zhang, César Augusto Fonticelha De Rose, Rajgopal Kannan, and Viktor K. Prasanna. Memmap: Compact and generalizable meta-lstm models for memory access prediction. In *Advances in Knowledge Discovery and Data Mining - 24th Pacific-Asia Conference*, volume 12085 of *PAKDD '20*, pages 57–68, Singapore, May 2020. Springer. doi:10.1007/978-3-030-47436-2\_5.
- 24 Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *2nd International Conference on Learning Representations, ICLR '14*, Banff, AB, Canada, April 2014. arXiv: 1312.6199.
- 25 Alexander Wei. Better and simpler learning-augmented online caching. In *Approximation, Randomization, and Combinatorial Optimization: Algorithms and Techniques*, volume 176 of *APPROX/RANDOM '20*, pages 60:1–60:17, Virtual Conference, August 2020. Schloss Dagstuhl - Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.APPROX/RANDOM.2020.60.
- 26 William A. Wulf and Sally A. McKee. Hitting the memory wall: implications of the obvious. *SIGARCH Computer Architecture News*, 23(1):20–24, 1995. doi:10.1145/216585.216588.

# Differentially Oblivious Turing Machines

Ilan Komargodski<sup>1</sup> 

The Hebrew University in Jerusalem, Israel

NTT Research, Palo Alto, CA, USA

<https://www.cs.huji.ac.il/~ilank/>

ilank@cs.huji.ac.il

Elaine Shi

Cornell University, Ithaca, NY, USA

Carnegie Mellon University, Pittsburgh, PA, USA

<http://elaineshi.com/>

runting@gmail.com

---

## Abstract

Oblivious RAM (ORAM) is a machinery that protects any RAM from leaking information about its secret input by observing only the access pattern. It is known that every ORAM must incur a logarithmic overhead compared to the non-oblivious RAM. In fact, even the seemingly weaker notion of differential obliviousness, which intuitively “protects” a single access by guaranteeing that the observed access pattern for every two “neighboring” logical access sequences satisfy  $(\epsilon, \delta)$ -differential privacy, is subject to a logarithmic lower bound.

In this work, we show that any *Turing machine* computation can be generically compiled into a differentially oblivious one with only *doubly* logarithmic overhead. More precisely, given a Turing machine that makes  $N$  transitions, the compiled Turing machine makes  $O(N \cdot \log \log N)$  transitions in total and the physical head movements sequence satisfies  $(\epsilon, \delta)$ -differential privacy (for a constant  $\epsilon$  and a negligible  $\delta$ ). We additionally show that  $\Omega(\log \log N)$  overhead is necessary in a natural range of parameters (and in the balls and bins model).

As a corollary, we show that there exist natural data structures such as stack and queues (supporting online operations) on  $N$  elements for which there is a differentially oblivious implementation on a Turing machine incurring amortized  $O(\log \log N)$  overhead per operation, while it is known that any oblivious implementation must consume  $\Omega(\log N)$  operations unconditionally even on a RAM. Therefore, we obtain the first *unconditional* separation between obliviousness and differential obliviousness in the most natural setting of parameters where  $\epsilon$  is a constant and  $\delta$  is negligible. Before this work, such a separation was only known in the balls and bins model. Note that the lower bound applies in the RAM model while our upper bound is in the Turing machine model, making our separation stronger.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Turing machines

**Keywords and phrases** Differential privacy, Turing machines, obliviousness

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.68

**Funding** *Ilan Komargodski*: Supported by an Alon Young Faculty Fellowship and by an ISF grant (No. 1774/20).

*Elaine Shi*: Supported by an NSF grant (award number CNS-1601879), an Office of Naval Research Young Investigator Program Award, and a Packard Fellowship.

**Acknowledgements** We thank the reviewers of ITCS 2021 for their feedback. We thank Hubert Chan and Kai-Min Chung for useful discussions.

---

<sup>1</sup> Corresponding author.



## 1 Introduction

An oblivious RAM (ORAM), introduced in the seminal work of Goldreich and Ostrovsky [17, 30, 18], is a tool for “encrypting” the access pattern of any RAM so that it looks “unrelated” to the underlying data. It is known that any ORAM scheme must incur at least  $\Omega(\log N)$  overhead, where  $N$  is the size of the memory. This was first shown by Goldreich and Ostrovsky [18] in the balls and bins model<sup>2</sup> and assuming that no cryptographic assumptions are used. More recently, Larsen and Nielsen [25] proved the same lower bound without the two aforementioned restrictions but requiring the ORAM to support operations arriving in an online manner. In fact, a follow-up work by Jacob et al. [23] showed that  $\Omega(\log N)$  overhead is necessary even for obliviously implementing very specific data structures (as defined in [43]) such as stacks, queues, and more.

Apparently, logarithmic overhead is necessary even for implementing a RAM with a (seemingly) much weaker security guarantee than full obliviousness. Persiano and Yeo [34] considered the notion of differentially oblivious RAM,<sup>3</sup> a relaxation of ORAM that only protects individual operations by guaranteeing  $(\epsilon, \delta)$ -differential privacy for the observed access pattern of the RAM (see Section 2.3 for a formal definition).<sup>4</sup> Differential obliviousness was also studied in the context of *specific* functionalities by Chan et al. [7] and Beimel et al. [4]. It is shown that there are tasks for which obtaining differential obliviousness might be easier than full obliviousness. For instance, Chan et al. [7] show that there is a differentially oblivious algorithm for sorting  $N$  records according to a 1-bit key while maintaining the relative ordering of records with identical keys in time  $O(N \cdot \log \log N)$ ,<sup>5</sup> while [26] showed a conditional  $\Omega(N \cdot \log N)$  lower bound for full fledged obliviousness *in the balls and bins model*. This leaves the following natural question open.

*Is there an unconditional separation between obliviousness and differential obliviousness?*

Let us remark that in the above question we are interested in the most standard models and range of parameters. For RAMs, we consider the standard word-RAM where each memory word is large enough to store its own logical address, where word-level addition and Boolean operations can be done in unit cost, and where the CPU has constant number of private registers. For  $(\epsilon, \delta)$ -differential obliviousness, we want schemes that are secure for  $\epsilon$  being a fixed constant and  $\delta$  being a negligible function. For Turing machines, we allow an arbitrary number (which is fixed as part of the machine’s description) of one-dimensional bi-directional infinite work tapes, where in every step the head can moved left, right, or stay in place.

<sup>2</sup> This model assumes that each memory word is “indivisible” and restricts the ORAM to only move blocks around and not apply any non-trivial encoding of the underlying secret data; see Boyle and Naor [6].

<sup>3</sup> Persiano and Yeo [34] called this notion differentially private RAM, but we prefer to use differentially oblivious RAM to (1) relate to the notion of oblivious RAM and stress that the goal is to preserve the physical access pattern’s privacy and (2) be aligned with previous work on the topic (Chan et al. [7]).

<sup>4</sup> Usually, differential privacy concerns the observed output of some algorithm. In our context, the output of an algorithm consists of the transcript of the computation: the physical memory accesses performed during the computation.

<sup>5</sup> Maintaining the relative ordering of records is called *stability*. Without stability, sorting records according to 1-bit keys is known to be doable (deterministically and obliviously) in linear time [2, 3].

## 1.1 Our Results

We present a large class of functionalities that can be made differentially oblivious with only  $O(\log \log N)$  overhead. The class includes many natural and useful algorithms and data structures such as stacks and queues and therefore implies an *unconditional* separation between obliviousness and differential obliviousness.

► **Theorem 1** (A separation; informal). *There exists a data structure (e.g., a stack or a queue) supporting  $N$  operations for which:*

1. *Any oblivious implementation (even on a RAM) requires  $\Omega(N \cdot \log N)$  operations;*
2. *There is an  $(\epsilon, \delta)$ -differentially oblivious two-tape Turing machine (defined below) that requires*

$$O(N \cdot (\log(1/\epsilon) + \log \log N + \log \log(1/\delta)))$$

*operations.*

*In particular, letting  $\epsilon > 0$  be a constant and  $\delta = 2^{-\log^2 N}$  (which is negligible), the number of operations incurred by the differentially oblivious machine is  $O(N \cdot \log \log N)$ .*

The above theorem follows from a much more general result about differentially oblivious Turing machines. Oblivious Turing machines were first introduced in 1979 by Pippenger and Fischer [35]. In this model, “memory accesses” correspond to the head’s movements throughout the execution of the algorithm (i.e., Left, Right, or Stay). Pippenger and Fischer showed how any multi-tape Turing machine can be *obliviously* simulated by a two-tape Turing machine with a *logarithmic* slowdown in running time. More precisely, any Turing machine that makes  $N$  steps can be simulated obliviously while consuming  $O(N \cdot \log N)$  steps. The simulation is deterministic and perfectly oblivious: the same sequence of head movements is observed for any two inputs.

Adapting the notion of differential obliviousness to the Turing machine model, we show that any Turing machine that makes  $N$  steps can be simulated by a differentially oblivious machine while making only  $O(N \cdot \log \log N)$  steps. Here, neighboring sequences of head movements are ones where only one transition is different. For instance, the logical sequences of transitions {Left, Right, Left, Right} and {Left, Right, Right, Right} are neighboring.

► **Theorem 2** (A differentially oblivious Turing machine; see Theorem 11). *For any  $\epsilon, \delta > 0$ , any  $k$ -tape Turing machine that makes at most  $N$  steps can be simulated by an  $(\epsilon, \delta)$ -differentially oblivious machine with  $\max\{2, k\}$  tapes making  $O(N \cdot (\log(1/\epsilon) + \log \log N + \log \log(1/\delta)))$  steps.*

As above, letting  $\epsilon > 0$  be a constant and  $\delta = \delta(N)$  be a particular negligible function, the number of steps incurred by the differentially oblivious machine is  $O(N \cdot \log \log N)$ . We note that the constant hidden in the  $O$  notation depends only on the description size of the given Turing machine (i.e., its alphabet size, number of tapes, etc). Let us remark that the number of tapes we use is essentially optimal since even without any security requirements simulating a  $k$ -tape Turing machine for  $k \geq 3$  on a  $(k - 1)$ -tape one is not known to be possible with better than logarithmic overhead in steps (Hennie and Stearns [22]). Also, simulating a 2-tape machine on a single tape machine has *polynomial* overhead (Hartmanis and Stearns [20] for the upper bound and Hennie [21] for a lower bound).

Theorem 1 follows from Theorem 2 as follows. Consider (for instance) the stack data structure on  $N$  elements, supporting (“online”) PUSH and Pop operations. By Theorem 2 and using the fact that a stack can be implemented in linear time on a Turing machine,

there is an  $(\epsilon, \delta)$ -differentially oblivious Turing machine implementing it whose overhead is  $O(\log \log N)$  for a constant  $\epsilon$  and negligible  $\delta$ , as above. As mentioned, the logarithmic lower bound follows from Jacob et al. [23].

Lastly, we observe that a lower bound of Chan et al. [7] can be tweaked to show that our construction is essentially optimal by showing that  $\Omega(\log \log N)$  overhead is necessary for differential obliviousness in a natural range of parameters and in the balls and bins model.

► **Theorem 3** (A lower bound; see Theorem 13). *There exists an algorithmic task for which there is a Turing machine that on input of size  $N$  completes it in  $O(N)$  steps. On the other hand, for any  $0 < s \leq \sqrt{N}$ ,  $\epsilon > 0$ ,  $0 < \beta < 1$ , and  $0 \leq \delta \leq \beta \cdot (\epsilon/s) \cdot e^{-2\epsilon \cdot s}$ , any  $(\epsilon, \delta)$ -differentially oblivious implementation in the balls and bins model (even on a RAM) for this task must consume  $\Omega(N \cdot \log s)$  steps with probability  $1 - \beta$ .*

In particular, for a constant  $\epsilon > 0$  and  $s \geq \log^2 N$ , we can set  $\delta = 2^{-\Omega(\log^2 N)}$  (which is negligible) and get that  $\Omega(\log s) = \Omega(\log \log N)$  overhead is necessary. Note that if we want  $\delta = 2^{-N^{0.1}}$ , then the lower bound above says that the best we can hope for is  $\Omega(\log N)$  overhead. As mentioned, with logarithmic overhead we can actually get perfect obliviousness for any Turing machine [35].

## 1.2 Related Work

Goldreich and Ostrovsky [17, 18] showed that any RAM that uses a memory of size  $N$  and makes  $T$  accesses, can be made oblivious using only  $O(T \cdot \text{poly} \log N)$  accesses. The resulting RAM is probabilistic and obliviousness holds against polynomial-time distinguishers assuming the existence of one-way functions. The concept of oblivious RAM has inspired an immense amount of research. One line of work, focuses on applications of such compilers to cryptography and security, including applications in cloud computing, secure processor design, multi-party computation, and more (for example, [31, 38, 39, 5, 14, 36, 29, 15, 42, 16, 27, 45, 28, 44]). Another line of work, focuses on improving the overhead of the compiler [37, 24, 19, 8, 40, 41]. Only recently, a couple of works [32, 2] have resolved the problem by presenting a compiler whose overhead is  $O(\log N)$  (while still relying on one-way functions).

Patel et al. [33] considered the natural question of what kind of security can one hope for while limiting the overhead of a RAM simulation to constant. They show a construction of an  $(\epsilon, 0)$ -differentially oblivious RAM with  $O(1)$  overhead for  $\epsilon = O(\log N)$  and also assuming that the client can store  $\omega(\log N)$  records. They also proved a lower bound which quantitatively improves upon the one of [34] in the dependence on  $\epsilon$  but is qualitatively worse since it is in the balls and bins model. Throughout this work, we focus on the setting where  $\epsilon$  is a fixed constant and also that the client's storage is a constant number of blocks.

The work of Pippenger and Fischer [35] came in a long line of works trying to pin down the exact relation between various different computational models. One notable work is that of Hennie and Stearns [22] who showed that any multi-tape Turing machine can be simulated by a two-tape machine with *logarithmic* overhead. Pippenger and Fischer's result can be viewed as a similar compiler except that their resulting machine is also oblivious. Note that the result of [22] is the reason why one should not hope to improve the number of tapes in the resulting machine in Theorem 2 to two (as this task, even without privacy, is not known to be possible with less than logarithmic overhead). Simulating a 2-tape Turing machine on a single tape machine requires polynomial overhead due to Hartmanis and Stearns [20] and Hennie [21].



Some of our ideas in the differentially oblivious Turing machine construction are reminiscent of the aforementioned differentially oblivious algorithm (in the RAM model) for *stable* tight compaction due to Chan et al. [7]. Technically, their algorithm uses similar tools from the differential privacy literature (namely, differentially private prefix sums due to Chan et al. and Dwork et al. [9, 10, 12]) but the way they use it differ in nature from our approach. Partly, this is because our target machine is a Turing machine rather than a RAM, and therefore, standard building blocks such as oblivious sorting (which they use) are inapplicable. Second, even if we allow compiling a Turing machine to a differentially oblivious RAM (rather than insisting on Turing machine as the target machine), we still cannot directly use their techniques for constructing stable tight compaction because their techniques which rely on oblivious sorting are *offline* in nature; and thus not compatible with the *online* nature of our differentially oblivious simulation.

### 1.3 Technical Roadmap

In this overview we will focus on simulating a Turing machine with a single tape for  $N$  steps. There are many complications and technical difficulties that arise in the multi-tape case, but we refer to the technical sections for details.

Given a Turing machine our goal is to hide the location of the head during the execution of the machine, in a differentially private manner. To this end, we first develop an efficiently *differentially private* algorithm for estimating the location of the head at pre-defined points in time. Naively, we could add a fresh Laplacian noise every time we need an estimate, but this will incur at least  $\sqrt{N}$  loss in the privacy budget (by standard composition theorems).

To get around this, inspired by the work of Chan et al. [7], we use a differentially private prefix sum algorithm [9, 10, 12] to account for the location of the head. Recall that in the prefix sum algorithm, a stream of number arrives in an online manner and the algorithm outputs the sum of all number seen so far, after seeing every number. We set up the numbers to correspond to head movements (“Left” for -1 and “Right” for 1) and show that this approach incurs only  $\text{poly log } N$  loss in privacy budget, which is good enough in terms of privacy. One challenge that we run into is that we need to implement the differentially private prefix sum algorithm on a Turing machine. It turns out that every time we need to get an estimate of the head’s location (i.e., get a prefix sum), we need to pay some non-trivial factor in running time and so we need to minimize the number of such estimations. Therefore, we design our algorithm to work with only one estimate of the head’s location every  $\text{poly log } N$  steps and amortize the cost of this estimation while processing the next  $\text{poly log } N$  steps of the Turing machine.

Once we have a good-enough estimate of the head’s location every  $\text{poly log } N$  steps, all that is left is to copy the nearby positions to a smaller *oblivious* Turing machine which we use to simulate the next  $\text{poly log } N$  steps. We set up the parameters in such a way that we copy enough positions around the estimated head’s location to actually include the *real* head position along with the relevant tape around it to perform the next  $\text{poly log } N$  steps so the above is well defined. The oblivious Turing machine that we need must provide an “initialization” procedure that allow us to start an oblivious Turing machine from an existing memory, and a “destruction” procedure which allows us to extract the memory to its original structure in the end of the execution. Pippenger and Fischer’s [35] construction does not provide such procedures so we describe a variant that does. As an independent contribution, our new oblivious Turing Machine is described in a language that more closely resembles the hierarchical oblivious RAM construction of Goldreich and Ostrovsky [17, 18] so it might be easier to understand for those who are familiar with the latter.



Lastly, let us remark that the above description was very high level and glossed over many technical details. For example, one technicality arises because we have to use the tapes sparingly in the compiled Turing machine to get a theorem that is tight in the number of tapes. To achieve this, we develop algorithmic tricks that allow us to reuse the same tape for multiple purposes without incurring any overhead in asymptotic running time. Another technical challenge arises because unlike earlier explorations on differentially obliviousness [7, 4], our target machine is a Turing machine rather than a RAM. This imposes additional constraints for our algorithm design, since we cannot use common building blocks such as oblivious sorting. Moreover, the *online* nature of our differentially oblivious simulation also renders some previous building blocks inadequate (which we discuss more in the Related Work section). We refer to the technical sections for details.

## 2 Preliminaries

For an integer  $n \in \mathbb{N}$  we denote by  $[n]$  the set  $\{1, \dots, n\}$ . A function  $\text{negl}: \mathbb{N} \rightarrow \mathbb{R}^+$  is *negligible* if for every constant  $c > 0$  there exists an integer  $N_c$  such that  $\text{negl}(\lambda) < \lambda^{-c}$  for all  $\lambda > N_c$ .

### 2.1 Turing Machines

We follow the presentation of Arora and Barak [1] for the definition of a  $k$ -tape Turing machine. A tape is an infinite bi-directional line of cells, each of which can hold a symbol from a finite set called the alphabet. Each tape is associated to a tape head that can potentially read or write symbols to the tape one cell at a time. The machine's computation is divided into discrete time steps, and the head can either stay in place or move left or right one cell in each step. More formally, a Turing machine  $M$  is described by a tuple  $(\Gamma, Q, \delta)$ , where  $\Gamma$  is a set of symbols that  $M$ 's tapes can contain,  $Q$  is the set of  $M$ 's possible states, and  $\delta: Q \times \Gamma^k \rightarrow Q \times \Gamma^k \times \{L, S, R\}^k$  is  $M$ 's transition function.

If the machine is in state  $q \in Q$  and  $(\sigma_1, \sigma_2, \dots, \sigma_k)$  are the symbols currently being read in the  $k$  tapes, and  $\delta(q, (\sigma_1, \dots, \sigma_k)) = (q', (\sigma'_1, \dots, \sigma'_k), z)$ , where  $z \in \{L, S, R\}^k$ , then at the next step the  $\sigma$  symbols in the  $k$  tapes will be replaced by the  $\sigma'$  symbols, the machine will be in state  $q'$ , and the  $k$  heads will move Left, Right, Stay in place, as given by  $z$ . There are additionally a read-only tape for the input and a write-only tape for the output, and perhaps a randomness tape if needed, but we ignore those when counting the number of tapes and only account for the work tapes.

The definition above is quite robust to the choices one makes regarding the alphabet size, the number of tapes, etc, since they are all equivalent in terms of complexity up to small factors. We recall the known facts which can be found, for example, in Arora and Barak [1].

► **Fact 4.** *It holds that:*

1. *Every function  $f$  that is computable in time  $T$  using alphabet  $\Gamma$ , can be computed in time  $O(\log |\Gamma| \cdot T)$  using an alphabet of size  $O(1)$ .*
2. *Every function  $f$  that is computable in time  $T$  using  $k$  tapes, can be computed in time  $O(k \cdot T^2)$  on a single tape machine and in time  $O(k \cdot T \cdot \log T)$  on a two-tape machine.*
3. *Every function  $f$  that is computable in time  $T$  using  $k$  bi-directional tapes, can be computed in time  $O(T)$  using  $k$  standard (uni-directional) tapes.*
4. *Every function  $f$  that is computable in time  $T$  using  $k$  tapes, can be computed in time  $k \cdot T$  using  $k$  tapes such that in each step only one of the tapes moves.*

We mention the dependence on  $k$  in the above terms for explicitness even though it is a constant.

In this work, we care about logarithmic factors so, by default, our Turing machine model is that of a two-tape machine. By the above, it does not matter if we consider uni-directional or bi-directional tapes. Constant factors in the alphabet size do not matter as well. All of the above only affect the constants which are hidden inside the  $O$  notation.

## 2.2 Differential Privacy

Differential privacy, introduced by Dwork et al. [11], is a property of algorithms that, very roughly, guarantees “security” for a single record in the input. Namely, if the algorithm acts on the information of a set of individuals, from the output it is hard to decide whether a particular individual’s information was used in the computation. This is formalized as follows. Let  $A$  be a probabilistic algorithm that takes as input a dataset. Let  $\text{Im}(A)$  be the set of all possible outputs of  $A$ . The algorithm  $A$  is said to be  $(\epsilon, \delta)$ -differentially private if for all datasets  $D_0$  and  $D_1$ , that differ only on one entry, and all possible subsets  $S \subseteq \text{Im}(A)$ , it holds that

$$\Pr[A(D_0) \in S] \leq e^\epsilon \cdot \Pr[A(D_1) \in S] + \delta,$$

where  $e$  is the base of the natural logarithm.

We refer to Dwork and Roth [13] for more information on differential privacy.

## 2.3 (Differentially) Oblivious Turing Machines

**Obliviousness.** Obliviousness is nowadays usually defined for RAMs and it guarantees that the access pattern of the RAM is “independent” of the underlying input. More specifically, given a RAM  $M$  and an input  $\mathbf{I}$ , we consider a random variable  $\text{Accesses}(M, \mathbf{I})$  that corresponds to the ordered sequence of memory locations  $M$  accesses during an execution on input  $\mathbf{I}$ . We then require that the distribution of  $\text{Accesses}(M, \mathbf{I}_1)$  is indistinguishable from  $\text{Accesses}(M, \mathbf{I}_2)$  for any  $\mathbf{I}_1$  and  $\mathbf{I}_2$  of the same length. The precise notion of indistinguishability can be either computational, statistical, or perfect, depending on the context.

A Turing machine can be thought of as a restricted version of RAM where random accesses are not allowed but any two consecutive accessed addresses must be to adjacent locations (i.e., the head can move at most one cell at a time). Adapting the notion of obliviousness to the Turing machine model requires that the tape’s head movements during the execution of the algorithm to not leak information about the inputs. Again one can define various notions of obliviousness, including computation, statistical, or perfect. We consider the strong notion of *deterministic* perfect obliviousness.

► **Definition 5** (Oblivious Turing machine). *A Turing machine  $M$  is said to be oblivious if for every input  $x \in \{0, 1\}^*$  and  $i \in [N]$ , the location of each of  $M$ ’s heads at the  $i$ th step of execution on input  $x$  is only a function of  $|x|$  and  $i$ .*

**Differential obliviousness.** Differential obliviousness was introduced by Chan et al. [7] as a relaxation of obliviousness for RAMs. At a high level, this security notion only protects individual operations, rather than the whole sequence of operations. This is formalized by requiring  $(\epsilon, \delta)$ -differential privacy for the observed access pattern of the RAM. In this case, two sequences of accesses  $\mathbf{I}_0$  and  $\mathbf{I}_1$  are neighboring if they are of the same length and differ in exactly one location accessed.

When we adapt the notion to the Turing machine model, one needs to distinguish between the input of the computation and the induced sequence of head movements that this input causes. While in some cases the two are analogous (which is the case in some of our results), in other cases there might be a gap. Namely, there are cases where the inputs are very close (in say, Hamming distance) and yet the resulting head movements are far from each other. Nevertheless, defining neighboring inputs w.r.t the observed sequence of head movements, as we do next, still implies a privacy guarantee for the inputs using standard group privacy theorems [11].

► **Definition 6** (Neighboring inputs). *Let  $M$  be a  $k$ -tape Turing machine. For  $\mathbf{J} \in \{0, 1\}^*$ , let  $\text{Movements}(M, \mathbf{J}) \in (\{L, R, S\}^k)^*$  be the sequence of head movements that  $M$  does on input  $\mathbf{J}$ . Two inputs  $\mathbf{J}_0, \mathbf{J}_1 \in \{0, 1\}^*$  are called neighboring if*

1.  $|\text{Movements}(M, \mathbf{J}_0)| = |\text{Movements}(M, \mathbf{J}_1)|$  and
  2.  $\text{Movements}(M, \mathbf{J}_0)$  and  $\text{Movements}(M, \mathbf{J}_1)$  differ in exactly one location.
- That is, letting  $(\ell_1^{b,1}, \dots, \ell_k^{b,1}), \dots, (\ell_1^{b,N}, \dots, \ell_k^{b,N}) = \text{Movements}(M, \mathbf{J}_b)$  for  $b \in \{0, 1\}$ , there is exactly one pair  $(i^*, j^*) \in [N] \times [k]$  such that  $\ell_{j^*}^{0,i^*} \neq \ell_{j^*}^{1,i^*}$  while for all other  $(i, j) \in [N] \times [k] \setminus \{(i^*, j^*)\}$ , it holds that  $\ell_j^{0,i} = \ell_j^{1,i}$ .*

Given this notion of neighboring inputs, we give the definition of differential obliviousness.

► **Definition 7** ( $(\epsilon, \delta)$ -differentially oblivious Turing machine). *A Turing machine  $M$  satisfies  $(\epsilon, \delta)$ -differential privacy iff for any neighboring inputs  $\mathbf{J}_0$  and  $\mathbf{J}_1$  and any set  $S \in (\{L, R, S\}^k)^*$  of possible sequence of head movements, it holds that*

$$\Pr[\text{Movements}(M, \mathbf{J}_0) \in S] \leq e^\epsilon \cdot \Pr[\text{Movements}(M, \mathbf{J}_1) \in S] + \delta.$$

### 3 Estimating Heads' Locations

In this section, we present an algorithm running on a Turing machine that outputs estimates to the location of the heads in a Turing machine computation. More precisely, the input to the algorithm is a sequence of movements of the heads of the machine (i.e., Left, Right or Stay for each tape), and it outputs an estimate to the location of the head in a-priori fixed intervals of time in an online fashion. The algorithm (1) outputs an estimate which is not too far from the true position of the head, (2) the estimates is differentially private, (3) the algorithm's head movements themselves are oblivious (i.e., data-independent), and (4) the algorithm is very efficient. The intervals at which we output an estimate on the location of the heads are denoted  $p$ .

► **Theorem 8.** *There exists an algorithm  $\text{EstimateHead}_{\epsilon, \delta}$  such that for any  $\epsilon, \delta > 0$ , the following holds. Fix any stream  $\mathbf{a} = a_1, a_2, \dots, a_N \in \{L, S, R\}^k$  that corresponds to the movements of the heads of a  $k$ -tape Turing machine. Let  $\sigma_i = \sum_{j=1}^{i \cdot p} a_j$  for  $i \in [N/p]$  (i.e., the true position of the heads every  $p$  steps). Let  $\{\tilde{\sigma}_i\}_{i \in [N/p]}$  denote a possible output of the algorithm  $\text{EstimateHead}_{\epsilon, \delta}$  when fed  $\mathbf{a}$  as an input in an online fashion. It holds that:*

1. **Utility:** *With probability 1 over the randomness of  $\text{EstimateHead}_{\epsilon, \delta}$ , it holds that*

$$\max_{i \in [N/p]} |\tilde{\sigma}_i - \sigma_i| \in O\left((1/\epsilon) \cdot \log^{1.5} N \cdot \log(1/\delta)\right).$$

2. **Differential privacy:**  $\text{EstimateHead}_{\epsilon, \delta}$  is  $(\epsilon, \delta)$ -differentially private. Here, neighboring sequences are defined in the natural way by allowing only one of the  $k$  indices in one of the  $N$   $a_i$ 's to differ between the two sequences.
3. **Obliviousness:** *The algorithm itself is perfectly oblivious.*
4. **Efficiency:** *The algorithm runs in time  $O(k \cdot N + k \cdot (N/p) \cdot (\log N))$ .*

**Proof.** The algorithm builds on the differentially private prefix-sum algorithm of Chan et al. [9, 10] and Dwork et al. [12]. Their algorithms address the problem of continuously estimating the prefix sums of elements in a given stream of numbers while maintaining differential privacy. We follow the presentation of these algorithm from Dwork and Roth [13, §12.3]. The algorithm is given a stream of numbers  $\mathbf{b} = b_1, b_2, \dots, b_N \in \{-1, 0, 1\}$  that the algorithm sees in an online fashion. The algorithm outputs, after seeing  $b_1, \dots, b_j$  an approximation of  $\sum_{i=1}^j b_i$ . This task is almost what we need to prove our theorem for  $k = 1$  (i.e., the machine has one tape). Indeed, a movement left (resp. right) can be interpreted as -1 (resp. 1) and staying in place corresponds to seeing 0. The location of the head is exactly the sum of those numbers. Additionally, it is not hard to observe that their algorithm is in fact oblivious (see below). Nevertheless, the running time of their algorithm on a Turing machine is  $O(N \cdot \log N)$  (see below). So, we need to (1) extend the algorithm to handle any  $k \geq 1$  tapes and (2) show how to implement it in the specified running time on a Turing machine. Since both goals are somewhat non-trivial to achieve, let us first briefly recall their algorithm and state its guarantees, and then describe our modifications.

Assume that  $N$  is a power of 2 (for simplicity and without loss of generality). We associate the  $N$  numbers to leaves of a full binary tree and then label each node in the tree with an “interval”. The  $i$ th leaf (for  $i \in [N]$ ) is labeled with  $[i, i]$ . An internal node is labeled with the interval that is the union of the intervals associated with its children. Now, with each node, labeled  $[s, t]$  in this tree, we associate a noisy count that approximates the sum of the values seen in positions  $s, s+1, \dots, t$  by adding noise from the appropriate distribution. In [9, 10, 12] the added noise was sampled from  $\text{Lap}((1 + \log_2 N)/\epsilon)$ , where  $\text{Lap}(s)$  denotes the (continuous) Laplace distribution with mean 0 and variance  $2s^2$ . To output  $\tilde{\sigma}_i$  (i.e., the approximation of  $\sum_{j=1}^i a_j$ ), we write  $i$  in binary to find at most  $\log_2 N$  intervals whose union is  $[1, i]$ , and compute the sum of the corresponding noisy counts. These intervals are associated to the nodes which are called *the frontier*. This algorithm satisfies  $(\epsilon, 0)$ -differential privacy and satisfies the following utility property: With probability  $1 - \delta$  over the randomness of the algorithm,

$$\max_{i \in [N/p]} |\tilde{\sigma}_i - \sigma_i| \leq O\left((1/\epsilon) \cdot \log^{1.5} N \cdot \log(1/\delta)\right).$$

It is easy to turn the utility property to be satisfied with probability 1 by outputting the exact prefix sum in the clear whenever the error in the output is too large. This causes the algorithm to be  $(\epsilon, \delta/2)$ -differentially private, as needed.

**Handling multiple tapes.** We extend the algorithm to handle  $k$  tapes by maintaining  $k$  prefix sums computed in parallel. This clearly does not hurt utility or obliviousness and only incurs a  $k$  factor in running time. However, naively, it incurs a  $k$  factor in differential privacy. Nevertheless, we observe that considering any two neighboring sequences of inputs,  $k - 1$  of the tapes will have the exact same access pattern while only one will differ in one position, and so this extension, in fact, does not incur a  $k$  factor in differential privacy.

**Running time.** The main challenge is to maintain an updated version of the noisy counts associated to the nodes in the frontier. Recall that the frontier is of size  $\log_2 N + 1$ . Naively, with the above algorithm, computing the frontier at time  $i + 1$  from the frontier at time  $i$  may cost up to  $O(\log N)$  work which is too expensive for us. However, recall that we do not need a prefix sum after every  $a_i$ , but rather we want to output one after every  $p$  inputs. So, instead of having a full binary tree where the leaves correspond to each input, we consider a full binary tree where each leaf corresponds to a sequence of  $p$  inputs and it is labeled by

their sum. The depth of this tree is  $\log_2(N/p)$  and the point is that we need to compute the “next” frontier (which costs about  $\log_2 N$ ) only once every  $p$  operations, so the total cost is  $O(k \cdot (N/p) \cdot \log N)$  plus the time it takes to aggregate the sum itself which is  $O(k \cdot N)$ , as needed.  $\blacktriangleleft$

► **Remark 9 (Sampling from Lap).** We emphasize that the above algorithm assumes that a Turing machine is capable of sampling from  $\text{Lap}(\cdot)$  in  $O(1)$  time. This is assumed for simplicity of presentation. However, it is possible to efficiently compute an estimate of this distribution on a standard Turing machine. The cost of this approximation is small good enough to obtain (asymptotically) the same final result in Theorem 11. Therefore, the assumption being made in this section is without loss of generality in the context of our main result. See details in Appendix A.

## 4 Oblivious Turing Machines

A classical result by Pippenger and Fischer [35] shows that any Turing machine computation can be made perfectly oblivious (i.e., Definition 5) on a two-tape machine with amortized logarithmic overhead. More precisely, any Turing machine that makes at most  $N$  steps can be made perfectly oblivious while making  $O(N \cdot \log N)$  steps.

In our application we need an oblivious Turing machine which support two additional properties. The first, called “initialization”, is that one can initialize an oblivious Turing machine with a given memory (as opposed to starting off with an empty memory). The second, called “destruction”, returns the state of the memory in a linear fashion.

Our construction is similar in spirit to the one of Pippenger and Fischer [35]. However, we present it in a language that more closely resembles the hierarchical oblivious RAM construction of Goldreich and Ostrovsky [17, 18] so it might be easier to understand for those who are familiar with the latter.

► **Theorem 10 (Oblivious Turing machine, revisited).** *Any  $k$ -tape Turing machine that makes at most  $N$  steps can be executed obliviously on a two-tape Turing machine with  $O(N)$  space and with  $O(k \cdot N \cdot \log N)$  steps. Additionally, the machine supports initialization and destruction.*

**Proof.** We will present the main idea in the special case where the given Turing machine has only a single tape and the resulting machine will have  $\ell = \lceil \log N \rceil$  tapes. Later, we will explain how to handle multiple tape machines in the input and simulate them obliviously with just two tapes (at the same cost).

We have  $\ell$  tapes and sometimes we will refer to these tapes as “levels” (analogously to levels in [18]’s hierarchical ORAM construction). For each  $i \geq 1$ , level  $T_i$  is a tape that contains at most  $\ell_i \triangleq 2^i - 1$  elements and it is thought of as a cyclic buffer. That is, the element on the right of the  $(2^i - 1)$ th element in level  $T_i$  is the 1st one. Our construction will maintain the following invariant. Let  $p$  be the pointer to the current head location of the original Turing machine. At the end of every  $2^i$  steps,  $T_i$  stores the content of the tape at positions  $[p - 2^{i-1} : p + 2^{i-1}]$ .

According to this, level  $T_1$  will always store the content of the cell pointed to by the head, level  $T_2$  stores the content of cells  $p - 1, p, p + 1$ , and so on. Notice that the same cell may be part of several levels and not all of the values will be consistent with each other. The freshest copy of a cell is always in the  $T_i$  with the smallest  $i$  that contains the cell. Reading the value of the cell pointed to by the head or writing to that cell is done by reading or writing (respectively) directly to  $T_1$ .

For every  $i \geq 0$ , at the end of every  $2^i$  steps, the following reorganization steps are performed:

1.  $T_{i+1}$  writes its updated contents to  $T_{i+2}$ . This is done by making a pass over  $T_{i+2}$  and scanning over  $T_{i+1}$  as a cycle buffer, making a real write whenever needed and making a dummy write otherwise.
2.  $T_{i+1}$  copies the corresponding segment it ought to store from  $T_{i+2}$ . This is done, as above, by making a pass over  $T_{i+2}$  and scanning over  $T_{i+1}$  as a cycle buffer, making a real write whenever needed and making a dummy write otherwise.

Correctness follows immediately by description. Perfect obliviousness follows from the fact that the head's movements are deterministic and a-priori fixed. For efficiency, consider any sequence of  $N$  steps. A read or a write are done at a single operations cost by just accessing  $T_1$ . It remains to account for the cost of the reorganization steps. Note that the  $N$  operations are confined to levels  $T_i$  for  $i \leq \log N + 2$ . By description and recalling that the size of level  $T_i$  is  $2^i - 1$ , the total amount of steps performed by the oblivious Turing machine is bounded by

$$\sum_{i=1}^{\log N + 2} \left\lceil \frac{N}{2^{i-1}} \right\rceil \cdot O(|T_{i+2}|) \in O(N \cdot \log N).$$

We now explain how to remove the simplifying assumptions we had, the first being that the input machine has only one tape and the second being that the resulting oblivious machine uses  $\log N$  many tapes. Let us first handle the former, letting  $k$  be the number of tapes used by the input machine. We use an encoding trick. We encode the  $k$  tapes into a single tape by first placing all the first cells from each tape, then the second cell, and so on. Each “track” will have its own head marker. By the construction of the oblivious Turing machine, all the tracks can be processed simultaneously (recall that our head movement sequence is deterministic), incurring a  $k$  multiplicative factor.

Now, we explain how to modify our Turing machine to use only two tapes. As a first step, let us place the different levels one after the other on the single tape. Naively, this incurs a blowup in running time due to the reorganization steps. Indeed, in the reorganization steps, we need to scan two levels “in parallel” as cyclic buffers. The only way to do this with a single tape is by moving back and forth in the tape which is too expensive. This is where we will use the second tape. When such a “parallel” scan is needed, we will copy one of the levels to the second tape, do the “parallel” scan by scanning both tapes in parallel, and then copy it back. This only incurs a constant overhead.

**Initialization and destruction.** In our application, we will need to an oblivious Turing machine with two additional features so we explain how to implement them next. The first is that we need to support initialization with a given memory which might not necessarily be empty. We implement this by starting with an empty memory, as described above, and modifying the memory one element at a time. If the number of steps that we eventually perform on the Turing machine is about the size of the initial memory, the cost of this step will be amortized away.

The second feature is a destruction procedure which outputs the memory at the end of the computation in a linear fashion. This is not so immediate since our construction does not store the memory in a linear fashion. Recall that our construction satisfies that at the end of every  $2^i$  steps, the cells corresponding to the level  $T_i$  store the content of the tape at positions  $[p - 2^{i-1} : p + 2^{i-1}]$ . This means that if “destruct” is invoked after a power-of-2

many steps, the memory is stored exactly in the cells corresponding to some level and one can make one linear scan to extract those elements and put them one next to the other. If `destruct` is invoked after some other number of steps, we need to modify this procedure slightly by collecting the most updated memory values of each cell from the appropriate level (now, the most updated values are spread amongst different cells). This again can be done by a single scan. ◀

## 5 A Differentially Oblivious Turing Machine

In this section, we describe our transformation from any Turing machine into a differentially oblivious one.

► **Theorem 11.** *For any  $\epsilon, \delta > 0$ , any  $k$ -tape Turing machine that makes at most  $N$  steps and consumes  $S$  space can be transformed into an  $(\epsilon, \delta)$ -differentially oblivious  $\max\{2, k\}$ -tape Turing machine that makes  $O(N \cdot (\log(1/\epsilon) + \log \log N + \log \log(1/\delta)))$  steps and consumes  $O(S + (1/\epsilon) \cdot \log^2 N \cdot \log(1/\delta))$  space.*

The construction of the differentially oblivious Turing machine uses the oblivious Turing machine construction from Section 4 and the head's location estimation algorithm from Theorem 3. We will present the construction in steps. We first assume that the input machine uses only one tape and the resulting machine will use many tapes. Then, we will explain how to get rid of both simplifying assumptions and therefore obtain Theorem 11.

### 5.1 From One Tape to Four Tapes

Assume first that the given machine,  $M$ , uses only a single tape. We first present a construction that compiles  $M$  into a differentially oblivious Turing machine  $\text{dp}M$  with 4 tapes. Fix  $\epsilon, \delta > 0$  for the rest of this section.

**Tape allocation.** The resulting Turing machine,  $\text{dp}M$ , will consist of four tapes, numbered 1, 2, 3, and 4, in the following order:

1. One tape to simulate the input Turing machine computation (recall that we assumed that the input machine has only one tape).
2. Two tapes for running an oblivious Turing machine (according to Section 4).
3. One tape to compute differentially private head's location estimation algorithm (according to Section 3).

**The algorithm.** As mentioned, we use the oblivious Turing machine implementation from Section 4 but since its overhead is logarithmic (in the running time of the non-oblivious machine), we do not want to apply it directly on our machine. Instead, we are going to break down the computation of the original machine into epochs and invoke the oblivious machine only within epochs. Concretely, we split the computation of  $M$  into epochs of size

$$p \triangleq (1/\epsilon) \cdot \log^2 N \cdot \log(1/\delta).$$

Each such epoch will be executed in its own “fresh” oblivious Turing machine and so the overhead will only be a doubly logarithmic factor in  $N$ . Next, we explain how  $\text{dp}M$  works.



---

**Algorithm**  $\text{dp}M_{\epsilon,\delta}$ .

---

1. Set  $h_{\approx}^0 = 0$  be the initial approximate position of the head (it is equal to the real position).
  2. Break the  $T$ -step computation into epochs of  $p$  steps of computation. For epoch  $i = 1, \dots, N/p$ , do:
    - a. Copy an area of size  $4p + 1$  around  $h_{\approx}^{i-1}$ , namely  $[h_{\approx}^{i-1} - 2p, h_{\approx}^{i-1} + 2p]$  to the oblivious Turing machine (Theorem 10). Perform the next  $p$  steps of computation there. At the end of the epoch, copy the state of these  $4p + 1$  cells back to the main tape.
    - b. In parallel, keep track of the movements of the head and count the offset of the head compared to the previous location,  $h_{\approx}^{i-1}$ . At the end of the epoch, invoke the differentially private head's location estimation algorithm (Theorem 8) with privacy parameters  $\epsilon$  and  $\delta$  to update the location of the head  $h_{\approx}^i$ .
- 

► **Theorem 12.** *For any  $\epsilon, \delta > 0$  and given any single-tape Turing machine  $M$  that makes at most  $N$  steps and consumes  $S$  space, the 4-tape machine  $\text{dp}M_{\epsilon,\delta}$  is  $(\epsilon, \delta)$ -differentially oblivious, makes at most  $O(N \cdot (\log(1/\epsilon) + \log \log N + \log \log(1/\delta)))$  steps and consumes  $O(S + (1/\epsilon) \cdot \log^2 N \cdot \log(1/\delta))$  space.*

**Proof.** We first prove correctness, ignoring obliviousness. Consider any sequence of operations. At any point in time, the oblivious Turing machine contains  $2p + 1$  memory cells and performs all necessary operations within. For correctness, by description, it is enough to show that the  $p$  operations are indeed contained within those  $2p + 1$  cells. Indeed, for this to hold it is enough to argue that  $h_{\approx}^i$  is close enough to the real location of the head:  $h_{\approx}^i - 2p \leq h^i - p$  and  $h_{\approx}^i + 2p \geq h^i + p$ , where  $h^i$  is the true location of the head. In other words, we need to show that

$$|h_{\approx}^i - h^i| \leq p.$$

By the utility property of the head's location estimation algorithm (Theorem 8), we know that  $|h_{\approx}^i - h^i| \leq (1/\epsilon) \cdot \log^{1.5} N \cdot \log(1/\delta)$  (this is the upper bound on the additive error of each head's location estimation for every  $i$ ). Now, the above inequality follows by recalling that  $p = (1/\epsilon) \cdot \log^2 N \cdot \log(1/\delta)$ .

To prove  $(\epsilon, \delta)$ -differential obliviousness, consider any two sequences of operations  $\mathbf{I}_0$  and  $\mathbf{I}_1$  that differ at one operation. Consider the random variable  $\tilde{I}_b$  corresponding to the physical tape heads locations on input  $\mathbf{I}_b$  for  $b \in \{0, 1\}$ . Say the two sequences  $\mathbf{I}_0$  and  $\mathbf{I}_1$  differ in the  $i$ th operation and are otherwise identical. Then, the first  $i - 1$  operations result with identical distributions of head locations in both executions (as all the underlying building blocks are perfectly oblivious). The only difference is in the  $i$ th operation. There, the head's locations might differ due to a different distribution of the head's location estimation algorithm (Theorem 8). However, we are guaranteed that this algorithm is  $(\epsilon, \delta)$ -differentially private. The rest of the heads' movements are perfectly oblivious: the oblivious Turing machine is perfectly oblivious, the head's location estimation algorithm itself is perfectly oblivious, and the other operations that we do in the implementation of  $\text{dp}M$  are trivially oblivious.

Lastly, we analyze efficiency by counting the total amount of work and space required to handle any sequence  $N$  operations that consume  $S$  space. Step 2a costs  $O(p \cdot \log p)$  operations and space due to Theorem 10 (the rest of the operations can be implemented in  $O(p)$  time and space). Computing the differentially private head's location estimation in Step 2b takes overall  $O(N + (N/p) \cdot (\log N + \log(1/\delta))) < O(N)$  time due to Theorem 8 (i.e., constant

amortized work per access). Otherwise, simulating the original computation and accounting for the location of the head, requires  $O(N)$  work and space. Overall, over  $N$  operations, the total space is  $O(N + p \cdot \log p)$  and the work is bounded by  $O(N \cdot \log p)$ . Plugging in  $p = (1/\epsilon) \cdot \log^2 N \cdot \log(1/\delta)$  completes the proof.  $\blacktriangleleft$

## 5.2 From One Tape to Two Tapes

In this section we show how to obtain the same result as in the previous section (i.e., Theorem 12), except that our resulting Turing machine will only use two tapes (instead of four). One tape will be used for the simulation of the original Turing machine plus one of the tapes of the oblivious Turing machine and the other tape will be used to perform the head's location estimation algorithm and the other tape of the oblivious Turing machine.

Recall that the tapes used for the oblivious Turing machine, both consume about  $p$  space, but one interacts with the main tape (call it tape  $o\text{TM}_1$ ) and the other acts as a scratch pad and the values that are written there are never accessed outside of the oblivious Turing machine implementation (call it tape  $o\text{TM}_2$ ). We will merge tape  $o\text{TM}_2$  into the tape that simulates the original computation, and tape  $o\text{TM}_1$  to the tape that computes the prefix sums.

**Tape 1 (Main Tape).** This tape will consist of the main computation tape as well as a blank area which is used for tape  $o\text{TM}_2$  of the oblivious Turing machine. We use the fact that the required space for the oblivious Turing machine is  $O(p)$  cells and so we will maintain such a “space of blanks” which will not be too far from the real position of the head and will be used whenever a new oblivious Turing machine is instantiated. Let us denote by  $S_{o\text{TM}} = O(p)$  the space consumption of the oblivious Turing machine. Our first tape, the one that simulates the computation of the original Turing machine, will maintain the invariant that in distance  $S_{o\text{TM}}$  from the location of the approximate head  $h_{\approx}$  to the right, there are  $S_{o\text{TM}}$  blank cells (the last blank cell has distance  $2S_{o\text{TM}}$  from  $h_{\approx}$ ). If this invariant holds, then whenever an epoch begins, we can move to the blank area and use it as the oblivious Turing machine tape. At the end of the epoch, we can go back to where we were. Since we perform  $p$  operations inside the oblivious Turing machine, the amortized cost of moving back and forth is  $O(1)$  per operation which is what we need.

We thus need to explain how to maintain the above invariant. The idea is to move the blank area together with the location of the head once every epoch. Namely, once we update the approximate position of the head  $h_{\approx}$ , we will also move the blank area appropriately so that its distance from the new  $h_{\approx}$  is as we require. Moving the blank area, as above, can be done simply in time  $O(p)$  using a designated size  $O(p)$  space in the other tape (tape 2) – this is done by moving the area that needs to go to the blank area to tape 2 (and replacing it with blanks), and then copying by moving both heads “in parallel”.

**Tape 2 (Secondary Tape).** This tape will consist of three areas, each of size  $O(p)$ . One of these areas will be used for moving the blank area in tape 1, as we explained above. Another area is for the computation of the head's location estimation – this algorithm has state of size  $O(\log N) < O(p)$  (which contains a frontier of a tree of noisy sums per interval). The third part is for tape  $o\text{TM}_1$  of the oblivious Turing machine (which also uses  $O(p)$  space).

The first and second parts in this tape are accessed at the end of every epoch and some computation of length  $O(p)$  is performed on each of them (either updating the prefix sum or moving data around). Since each epoch handles  $O(p)$  operations, the amortized cost of this part is  $O(1)$  per operation of the original machine. The third part, in contrast, is accessed throughout the epoch, and there we get  $O(\log p)$  overhead per operation.

### 5.3 From $k$ Tapes to $k$ Tapes (for $k \geq 2$ )

This extension is done by making two changes. First we use  $k$  tapes to simulate the computation of the original  $k$ -tape machine (instead of just a single tape). Second, we use the algorithm for estimating the head's position which works for  $k$  tape machines (Theorem 8) – this incurs an overhead of  $k$  operations per step. Recall that this algorithm just runs the algorithm for estimating the head's position of a single tape  $k$  times (independently). It remains to explain where we execute this algorithm and also where we execute the oblivious Turing machine since now we do not have an extra work tape.

The idea is to first modify each tape to have *two* “blank areas”, each as above. Say that one blank area will be on the left of the head's position and the other one on the right. The one on the left will act as the “Main Tape” in the above construction and the one on the right as the “Secondary Tape” for another tape. Concretely, tape  $(i + 1) \bmod k$  acts as the “Secondary Tape” of tape  $i$  (for all  $i \in [k]$ ).

That is, the blank area on the left of the head of tape  $i$ , is used to simulate the computation of tape  $i$  in the original machine and also to execute tape  $\text{oTM}_2$  of the oblivious Turing machine when simulating tape  $i$  (this is exactly the same usage of the blank area as above). The blank area on the right of the head of tape  $i$  consists of three areas: (1) an area used to maintain the blank areas in tape  $(i + 1) \bmod k$ , (2) an area used for the computation of the head's location estimation of tape  $(i + 1) \bmod k$ , and (3) tape  $\text{oTM}_1$  of the oblivious Turing machine when simulating tape  $(i + 1) \bmod k$ . Overall, these changes incur an extra  $O(k)$  factor in the overhead of the simulation.

## 6 Lower Bound

In this section we prove that our differentially oblivious Turing machine is optimal in terms of overhead in a natural range of parameters. Specifically, we prove the following theorem.

► **Theorem 13.** *There exists an algorithmic task for which there is a Turing machine that on input of size  $N$  completes it in  $O(N)$  steps. On the other hand, for any  $0 < s \leq \sqrt{N}$ ,  $\epsilon > 0$ ,  $0 < \beta < 1$ , and  $0 \leq \delta \leq \beta \cdot (\epsilon/s) \cdot e^{-2\epsilon \cdot s}$ , any  $(\epsilon, \delta)$ -differentially oblivious implementation (even on a RAM and in the balls and bins model) for this task must consume  $\Omega(N \cdot \log s)$  steps with probability  $1 - \beta$ .*

**Proof.** In the work of Chan et al. [7] the following theorem concerning the required overhead to stably sort a set of balls according to associated 1-bit keys while maintaining differential obliviousness. Here, we assume that the balls are opaque and so no non-trivial encoding on them can be done [6].

► **Theorem 14** (Theorem 4.7 in [7]). *Let  $0 < s \leq \sqrt{N}$ . Suppose  $\epsilon > 0$ ,  $0 < \beta < 1$ , and  $0 \leq \delta \leq \beta \cdot (\epsilon/s) \cdot e^{-2\epsilon \cdot s}$ . Then, any (even randomized) stable sorting algorithm for balls according to associated 1-bit keys in the RAM model that is  $(\epsilon, \delta)$ -differentially oblivious must have some input, on which it incurs at least  $\Omega(N \cdot \log s)$  memory accesses with probability at least  $1 - \beta$ .*

The task of stably sorting  $N$  balls according to associated 1-bit keys can be implemented using a Turing machine in  $O(N)$  steps. Consider an input of the form  $(k_1, v_1), \dots, (k_N, v_N)$ , where  $k_i \in \{0, 1\}$  is a 1-bit key and  $v_i$  is the  $i$ th ball. The idea is to scan the input from the beginning and whenever we see an element  $(k_i, v_i)$  we do one of the following. If  $k_i = 0$ , we write  $(k_i, v_i)$  to the next position in the output tape. If  $k_i = 1$ , we write it to the next position in the work tape. After we finish scanning the input, we scan the output again and write all elements from first to last into the output tape. It is immediate that this algorithm is correct and has  $O(N)$  running time. ◀

## References

- 1 Sanjeev Arora and Boaz Barak. *Computational Complexity - A Modern Approach*. Cambridge University Press, 2009.
- 2 Gilad Asharov, Ilan Komargodski, Wei-Kai Lin, Kartik Nayak, Enoch Peserico, and Elaine Shi. OptORAMa: Optimal oblivious RAM. In *Advances in Cryptology - EUROCRYPT*, pages 403–432, 2020.
- 3 Gilad Asharov, Ilan Komargodski, Wei-Kai Lin, Enoch Peserico, and Elaine Shi. Oblivious parallel tight compaction. In *1st Conference on Information-Theoretic Cryptography, ITC*, pages 11:1–11:23, 2020.
- 4 Amos Beimel, Kobbi Nissim, and Mohammad Zaheri. Exploring differential obliviousness. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM*, pages 65:1–65:20, 2019.
- 5 Vincent Bindschaedler, Muhammad Naveed, Xiaorui Pan, XiaoFeng Wang, and Yan Huang. Practicing oblivious access on cloud storage: the gap, the fallacy, and the new way forward. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 837–849. ACM, 2015.
- 6 Elette Boyle and Moni Naor. Is there an oblivious RAM lower bound? In *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science, ITCS*, pages 357–368, 2016.
- 7 T.-H. Hubert Chan, Kai-Min Chung, Bruce M. Maggs, and Elaine Shi. Foundations of differentially oblivious algorithms. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 2448–2467. SIAM, 2019.
- 8 T.-H. Hubert Chan, Yue Guo, Wei-Kai Lin, and Elaine Shi. Oblivious hashing revisited, and applications to asymptotically efficient ORAM and OPRAM. In *Advances in Cryptology - ASIACRYPT*, pages 660–690, 2017.
- 9 T.-H. Hubert Chan, Elaine Shi, and Dawn Song. Private and continual release of statistics. In *Automata, Languages and Programming, 37th International Colloquium, ICALP*, pages 405–417, 2010.
- 10 T.-H. Hubert Chan, Elaine Shi, and Dawn Song. Private and continual release of statistics. *ACM Trans. Inf. Syst. Secur.*, 14(3):26:1–26:24, 2011.
- 11 Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam D. Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography, Third Theory of Cryptography Conference, TCC*, pages 265–284, 2006.
- 12 Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy N. Rothblum. Differential privacy under continual observation. In *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC*, pages 715–724. ACM, 2010.
- 13 Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3-4):211–407, 2014.
- 14 Christopher W Fletcher, Marten van Dijk, and Srinivas Devadas. A secure processor architecture for encrypted computation on untrusted programs. In *Proceedings of the seventh ACM workshop on Scalable trusted computing*, pages 3–8. ACM, 2012.
- 15 Christopher W. Fletcher, Ling Ren, Albert Kwon, Marten van Dijk, and Srinivas Devadas. Freecursive ORAM: [nearly] free recursion and integrity verification for position-based oblivious RAM. In *Proceedings of the Twentieth International Conference on Architectural Support for Programming Languages and Operating Systems, ASPLOS*, pages 103–116. ACM, 2015.
- 16 Craig Gentry, Shai Halevi, Charanjit Jutla, and Mariana Raykova. Private database access with he-over-oram architecture. In *International Conference on Applied Cryptography and Network Security*, pages 172–191. Springer, 2015.
- 17 Oded Goldreich. Towards a theory of software protection and simulation by oblivious rams. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, STOC*, pages 182–194, 1987.

- 18 Oded Goldreich and Rafail Ostrovsky. Software protection and simulation on oblivious RAMs. *J. ACM*, 1996.
- 19 Michael T. Goodrich and Michael Mitzenmacher. Privacy-preserving access of outsourced data via oblivious RAM simulation. In *Automata, Languages and Programming - 38th International Colloquium, ICALP*, pages 576–587, 2011.
- 20 Juris Hartmanis and Richard E Stearns. On the computational complexity of algorithms. *Transactions of the American Mathematical Society*, 117:285–306, 1965.
- 21 F. C. Hennie. One-tape, off-line turing machine computations. *Inf. Control.*, 8(6):553–578, 1965.
- 22 F. C. Hennie and Richard Edwin Stearns. Two-tape simulation of multitape turing machines. *J. ACM*, 13(4):533–546, 1966.
- 23 Riko Jacob, Kasper Green Larsen, and Jesper Buus Nielsen. Lower bounds for oblivious data structures. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 2439–2447. SIAM, 2019.
- 24 Eyal Kushilevitz, Steve Lu, and Rafail Ostrovsky. On the (in)security of hash-based oblivious RAM and a new balancing scheme. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 143–156, 2012.
- 25 Kasper Green Larsen and Jesper Buus Nielsen. Yes, there is an oblivious RAM lower bound! In *Advances in Cryptology - CRYPTO*, pages 523–542, 2018.
- 26 Wei-Kai Lin, Elaine Shi, and Tiancheng Xie. Can we overcome the  $n \log n$  barrier for oblivious sorting? In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 2419–2438, 2019.
- 27 Chang Liu, Xiao Shaun Wang, Kartik Nayak, Yan Huang, and Elaine Shi. OblivM: A programming framework for secure computation. In *IEEE Symposium on Security and Privacy*, 2015.
- 28 Steve Lu and Rafail Ostrovsky. Distributed oblivious RAM for secure two-party computation. In *Theory of Cryptography*, pages 377–396. Springer, 2013.
- 29 Martin Maas, Eric Love, Emil Stefanov, Mohit Tiwari, Elaine Shi, Krste Asanovic, John Kubiatowicz, and Dawn Song. PHANTOM: practical oblivious computation in a secure processor. In *2013 ACM SIGSAC Conference on Computer and Communications Security, CCS*, pages 311–324, 2013.
- 30 Rafail Ostrovsky. Efficient computation on oblivious rams. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing*, pages 514–523, 1990.
- 31 Rafail Ostrovsky and Victor Shoup. Private information storage (extended abstract). In *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing, STOC*, pages 294–303, 1997.
- 32 Sarvar Patel, Giuseppe Persiano, Mariana Raykova, and Kevin Yeo. Panorama: Oblivious RAM with logarithmic overhead. In *FOCS*, 2018.
- 33 Sarvar Patel, Giuseppe Persiano, and Kevin Yeo. What storage access privacy is achievable with small overhead? In *Proceedings of the 38th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS*, pages 182–199, 2019.
- 34 Giuseppe Persiano and Kevin Yeo. Lower bounds for differentially private rams. In *Advances in Cryptology - EUROCRYPT 2019*, pages 404–434, 2019.
- 35 Nicholas Pippenger and Michael J. Fischer. Relations among complexity measures. *J. ACM*, 26(2):361–381, 1979.
- 36 Ling Ren, Xiangyao Yu, Christopher W. Fletcher, Marten van Dijk, and Srinivas Devadas. Design space exploration and optimization of path oblivious RAM in secure processors. In *The 40th Annual International Symposium on Computer Architecture, ISCA*, pages 571–582. ACM, 2013.
- 37 Elaine Shi, T.-H. Hubert Chan, Emil Stefanov, and Mingfei Li. Oblivious RAM with  $O((\log N)^3)$  worst-case cost. In *Advances in Cryptology - ASIACRYPT*, pages 197–214, 2011.

- 38 Emil Stefanov and Elaine Shi. Oblivstore: High performance oblivious cloud storage. In *2013 IEEE Symposium on Security and Privacy*, pages 253–267. IEEE, 2013.
- 39 Emil Stefanov, Elaine Shi, and Dawn Xiaodong Song. Towards practical oblivious RAM. In *19th Annual Network and Distributed System Security Symposium, NDSS*, 2012.
- 40 Emil Stefanov, Marten van Dijk, Elaine Shi, Christopher W. Fletcher, Ling Ren, Xiangyao Yu, and Srinivas Devadas. Path ORAM: an extremely simple oblivious RAM protocol. In *ACM SIGSAC Conference on Computer and Communications Security, CCS*, pages 299–310, 2013.
- 41 Xiao Wang, T.-H. Hubert Chan, and Elaine Shi. Circuit ORAM: on tightness of the goldreich-ostrovsky lower bound. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, CCS*, pages 850–861, 2015.
- 42 Xiao Shaun Wang, Yan Huang, T.-H. Hubert Chan, Abhi Shelat, and Elaine Shi. SCORAM: oblivious RAM for secure computation. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 191–202, 2014.
- 43 Xiao Shaun Wang, Kartik Nayak, Chang Liu, T.-H. Hubert Chan, Elaine Shi, Emil Stefanov, and Yan Huang. Oblivious data structures. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, CCS*, pages 215–226, 2014.
- 44 Peter Williams, Radu Sion, and Alin Tomescu. Privatefs: A parallel oblivious file system. In *ACM Conference on Computer and Communications Security (CCS)*, 2012.
- 45 Samee Zahur, Xiao Shaun Wang, Mariana Raykova, Adria Gascón, Jack Doerner, David Evans, and Jonathan Katz. Revisiting square-root ORAM: efficient random access in multi-party computation. In *IEEE Symposium on Security and Privacy, S&P*, pages 218–234, 2016.

## A Sampling Noise on a Turing machine

One of the operations our differentially oblivious Turing machine needs to do is to sample from the (continuous) Laplacian distribution  $\text{Lap}(x)$ ; this is used in the algorithm for estimating the head’s location; see Section 3. There, we need to generate a sample from  $\text{Lap}((1 + \log_2 N)/\epsilon)$  and we need to do this about  $N$  times. Recall that a Laplacian distribution is unbounded and samples need infinite precision. We show that with small tolerable loss in precision (which does not affect our final result), one can sample an approximation from this distribution on a standard Turing machine.

We assume that  $\ln(1/\epsilon)$  is an integer so that we do not have rounding issues. Also, recall that  $\delta$  is a negligible function of the form  $\exp(-\log^2 N)$ . First, we switch to a bounded version of the distribution, chopping off the tail which contains elements that occur with negligible probability. We can assume that we sample from the range  $\pm(\log(N)/\epsilon) \cdot \text{poly} \log(1/\delta)$ . Let us call  $\delta_0$  the probability mass that we chopped off. Sampling from the bounded version turns our  $(\epsilon, \delta')$ -differentially private prefix sum algorithm into an  $(\epsilon, \delta)$ -differentially private one, where  $\delta = \delta' + N \cdot (\epsilon^\epsilon \cdot \delta_0 + \delta_0)$ . To see this, observe that we have essentially  $N$  instances of the  $\text{Lap}$  noise and so by a simple union bound, the statistical distance between each event w.r.t the bounded distribution happens with probability at most  $N \cdot \delta_0$  larger than in the unbounded version. Namely, for any set  $S$ ,  $\Pr[X_{\text{bounded}} \in S] \leq \Pr[X \in S] + N \cdot \delta_0$ , where  $X_{\text{bounded}}$  is the output of the mechanism when using bounded noise and  $X$  is the original mechanism. Then, by differential privacy,  $\Pr[X \in S] + N \cdot \delta_0 \leq e^\epsilon \Pr[Y \in S] + \delta' + N\delta_0$ , where  $Y$  is another arbitrary event sampled from the unbounded noise version. Then again by bounding the noise used in  $Y$ , we get

$$\Pr[X_{\text{bounded}} \in S] \leq e^\epsilon (\Pr[Y_{\text{bounded}} \in S] + N\delta_0) + \delta' + N\delta_0.$$

Since we think of  $\delta_0$  as being negligible in  $N$  and  $\epsilon$  being a constant,  $\delta$  is also negligible.<sup>6</sup>

<sup>6</sup> The above analysis was very loose. In particular, one can do a tighter analysis and not lose the linear-in- $N$  factor in  $\delta$  but for our purposes it does not matter since  $\delta$  is negligible in  $N$  anyway.



The next step is to represent each element in the bounded range with finite precision. We want to lose at most  $\delta$  factor in precision (which will add another additive  $\delta$  factor to our additive error), and so if we use  $\ell$  bits of precision, we have the inequality:

$$2^{-\ell} \cdot ((\log N)/\epsilon) \cdot \text{poly log}(1/\delta) \leq \delta.$$

This means that it is enough to use  $\ell \in O(\log((\log N \cdot \log \log(1/\delta))/(\epsilon\delta)))$  bits of precision which can be bounded by  $O(\log^2(1/\delta))$  bits since  $\delta$  is negligible in  $N$  and  $\epsilon$  is a constant. Therefore, all operations can be executed efficiently enough (in time  $O(\text{poly log } N)$ ), which by slightly changing parameters (e.g., the value of  $p$ ), does not affect our asymptotic upper bound on the running time of our differentially private Turing machine.





# Quantitative Correlation Inequalities via Semigroup Interpolation

Anindya De

University of Pennsylvania, Philadelphia, PA, USA  
anindyad@seas.upenn.edu

Shivam Nadimpalli

Columbia University, New York, NY, USA  
sn2855@columbia.edu

Rocco A. Servedio

Columbia University, New York, NY, USA  
ras2105@columbia.edu

---

## Abstract

Most correlation inequalities for high-dimensional functions in the literature, such as the Fortuin-Kasteleyn-Ginibre inequality and the celebrated Gaussian Correlation Inequality of Royen, are *qualitative* statements which establish that any two functions of a certain type have non-negative correlation. We give a general approach that can be used to bootstrap many *qualitative* correlation inequalities for functions over product spaces into *quantitative* statements. The approach combines a new extremal result about power series, proved using complex analysis, with harmonic analysis of functions over product spaces. We instantiate this general approach in several different concrete settings to obtain a range of new and near-optimal quantitative correlation inequalities, including:

- A quantitative version of Royen’s celebrated Gaussian Correlation Inequality [23]. In [23] Royen confirmed a conjecture, open for 40 years, stating that any two symmetric convex sets must be non-negatively correlated under any centered Gaussian distribution. We give a lower bound on the correlation in terms of the vector of degree-2 Hermite coefficients of the two convex sets, conceptually similar to Talagrand’s quantitative correlation bound for monotone Boolean functions over  $\{0, 1\}^n$  [26]. We show that our quantitative version of Royen’s theorem is within a logarithmic factor of being optimal.
- A quantitative version of the well-known FKG inequality for monotone functions over any finite product probability space. This is a broad generalization of Talagrand’s quantitative correlation bound for functions from  $\{0, 1\}^n$  to  $\{0, 1\}$  under the uniform distribution [26]; the only prior generalization of which we are aware is due to Keller [17, 15, 16], which extended [26] to product distributions over  $\{0, 1\}^n$ . In the special case of  $p$ -biased distributions over  $\{0, 1\}^n$  that was considered by Keller, our new bound essentially saves a factor of  $p \log(1/p)$  over the quantitative bounds given in [17, 15, 16]. We also give a quantitative version of the FKG inequality for monotone functions over the continuous domain  $[0, 1]^n$ , answering a question of Keller [16].

**2012 ACM Subject Classification** Mathematics of computing; Mathematics of computing → Probability and statistics

**Keywords and phrases** complex analysis, correlation inequality, FKG inequality, Gaussian correlation inequality, harmonic analysis, Markov semigroups

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.69

**Related Version** Full Version: <https://arxiv.org/abs/2012.12216>

**Funding** *Anindya De*: A.D. is supported by NSF grants CCF 1910534 and CCF 1926872.

*Shivam Nadimpalli*: S.N. is supported by NSF grants CCF-1563155 and by CCF-1763970.

*Rocco A. Servedio*: R.A.S. is supported by NSF grants CCF-1814873, IIS-1838154, CCF-1563155, and by the Simons Collaboration on Algorithms and Geometry.



© Anindya De, Shivam Nadimpalli, and Rocco A. Servedio;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 69; pp. 69:1–69:20



Leibniz International Proceedings in Informatics

LIPIcs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

## 1 Introduction

*Correlation inequalities* are theorems stating that for certain classes of functions and certain probability distributions  $\mathcal{D}$ , any two functions  $f, g$  in the class must be non-negatively correlated with each other under  $\mathcal{D}$ , i.e. it must be the case that  $\mathbf{E}_{\mathcal{D}}[fg] - \mathbf{E}_{\mathcal{D}}[f] \mathbf{E}_{\mathcal{D}}[g] \geq 0$ . Inequalities of this type have a long history, going back at least to a well-known result of Chebyshev, “Chebyshev’s order inequality,” which states that for any two nondecreasing sequences  $a_1 \leq \dots \leq a_n$ ,  $b_1 \leq \dots \leq b_n$  and any probability distribution  $p$  over  $[n] = \{1, \dots, n\}$ , it holds that

$$\sum_{i=1}^n a_i b_i p_i \geq \left( \sum_{i=1}^n a_i p_i \right) \left( \sum_{i=1}^n b_i p_i \right).$$

Modern correlation inequalities typically deal with high dimensional rather than one dimensional functions. Results of this sort have proved to be of fundamental interest in many fields such as combinatorics, analysis of Boolean functions, statistical physics, and beyond.

Perhaps the simplest high-dimensional correlation inequality is the well known Harris-Kleitman theorem [10, 19], which states that if  $f, g : \{0, 1\}^n \rightarrow \{0, 1\}$  are monotone functions (meaning that  $f(x) \leq f(y)$  whenever  $x_i \leq y_i$  for all  $i$ ) then  $\mathbf{E}[fg] - \mathbf{E}[f] \mathbf{E}[g] \geq 0$ , where expectations are with respect to the uniform distribution over  $\{0, 1\}^n$ . The Harris-Kleitman theorem has a one-paragraph proof by induction on  $n$ ; on the other end of the spectrum is the Gaussian Correlation Inequality (GCI), which states that if  $K, L \subseteq \mathbb{R}^n$  are any two symmetric convex sets and  $\mathcal{D}$  is any centered Gaussian distribution over  $\mathbb{R}^n$ , then  $\mathbf{E}_{\mathcal{D}}[KL] - \mathbf{E}_{\mathcal{D}}[K] \mathbf{E}_{\mathcal{D}}[L] \geq 0$  (where we identify sets with their 0/1-valued indicator functions). This was a famous conjecture for four decades before it was proved by Thomas Royen in 2014 [23]. Other well-known correlation inequalities include the Fortuin-Kasteleyn-Ginibre (FKG) inequality [7], which is an important tool in statistical mechanics and probabilistic combinatorics; the Griffiths-Kelly-Sherman (GKS) inequality [9, 18], which is a correlation inequality for ferromagnetic spin systems; and various generalizations of the GKS inequality to quantum spin systems [8, 25].

### 1.1 Quantitative Correlation Inequalities

Here, we attempt to obtain *quantitative* correlation inequalities. Consider the following representative example: For two monotone Boolean functions  $f, g : \{0, 1\}^n \rightarrow \{0, 1\}$ , as discussed above, the Harris-Kleitman theorem states that  $\mathbf{E}[fg] - \mathbf{E}[f] \mathbf{E}[g] \geq 0$ . It is easy to check that the Harris-Kleitman inequality is tight if and only if  $f$  and  $g$  depend on disjoint sets of variables. One might therefore hope to get an improved bound by measuring how much  $f$  and  $g$  depend simultaneously on the same coordinates. Such a bound was obtained by Talagrand [26] in an influential paper (appropriately titled “How much are increasing sets correlated?”).

To explain Talagrand’s main result, we recall the standard notion of *influence* from the analysis of Boolean functions [21]. For a Boolean function  $f : \{0, 1\}^n \rightarrow \{0, 1\}$ , the *influence* of coordinate  $i$  on  $f$  is defined to be  $\mathbf{Inf}_i[f] := \Pr_{\mathbf{x} \sim U_n}[f(\mathbf{x}) \neq f(\mathbf{x}^{\oplus i})]$ , where  $U_n$  is the uniform distribution on  $\{0, 1\}^n$  and  $\mathbf{x}^{\oplus i}$  is obtained by flipping the  $i^{\text{th}}$  bit of  $\mathbf{x}$ . Talagrand proved the following *quantitative* version of the Harris-Kleitman inequality:

$$\mathbf{E}[fg] - \mathbf{E}[f] \mathbf{E}[g] \geq \frac{1}{C} \cdot \Psi \left( \sum_{i=1}^n \mathbf{Inf}_i[f] \mathbf{Inf}_i[g] \right) \quad (1)$$

■ **Table 1** Qualitative and quantitative correlation inequalities. Here  $\gamma$  denotes the standard Gaussian distribution  $\mathcal{N}(0, 1)^n$ ;  $\{0, 1\}_p^n$  denotes the  $p$ -biased hypercube (with no subscript corresponding to  $p = 1/2$ , i.e. the uniform distribution);  $\pi$  denotes any distribution over  $\{0, \dots, m-1\}$ ; and  $[0, 1]^n$  is endowed with the Lebesgue measure.

	Qualitative Bounds	Quantitative Bounds
Monotone $f, g \in L^2(\mathbb{R}^n, \gamma)$	[7]	[14]
Symmetric, convex $K, L \subseteq \mathbb{R}_\gamma^n$	[23]	Theorem 21
Convex $f, g \in L^2(\mathbb{R}^n, \gamma)$	[11]	Deferred to full version.
Monotone $f, g : \{0, 1\}^n \rightarrow \{0, 1\}$	[10, 19, 7]	[26]
Monotone $f, g : \{0, 1\}_p^n \rightarrow \mathbb{R}$	[7]	[15], Theorem 29
Monotone $f, g : \{0, \dots, m-1\}_\pi^n \rightarrow \mathbb{R}$	[7]	Theorem 29
Monotone $f, g \in L^2([0, 1]^n)$	[22]	Deferred to full version.

where  $\Psi(x) := x/\log(e/x)$ ,  $C > 0$  is an absolute constant, and the expectations are with respect to the uniform measure. A simple corollary of this result is that  $\mathbf{E}[fg] = \mathbf{E}[f]\mathbf{E}[g]$  if and only if the sets of influential variables for  $f$  and  $g$  are disjoint. In [26] itself, Talagrand gives an example for which Equation (1) is tight up to constant factors.

Talagrand’s result has proven to be influential in the theory of Boolean functions, and several works [15, 16, 17, 12] have obtained extensions and variants of this inequality for product distributions over  $\{0, 1\}^n$ . An analogue of Talagrand’s inequality in the setting of monotone functions over Gaussian space was obtained by Keller, Mossel and Sen [14] using a new notion of “geometric influences.” Beyond these results, we are not aware of *quantitative* correlation inequalities in other settings, even though (as discussed above) a wide range of *qualitative* correlation inequalities are known. In particular, even for very simple and concrete settings such as the solid cube  $[0, 1]^n$  endowed with the uniform measure or the  $m$ -ary cube  $\{0, 1, \dots, m-1\}^n$  with a product measure, no quantitative versions of the FKG inequality were known (see the discussion immediately following Theorem 4 of Keller [13]). As a final example, no quantitative version of the Gaussian Correlation Inequality was previously known.

## 1.2 Our Contributions

We establish a general framework to transfer qualitative correlation inequalities into quantitative correlation inequalities. We apply this general framework to obtain a range of new quantitative correlation inequalities, which include the following:

1. Quantitative versions of Royen’s Gaussian Correlation Inequality and Hu’s correlation inequality [11] for symmetric convex functions over Gaussian space;
2. A quantitative FKG inequality for a broad class of product distributions, including arbitrary product distributions over finite domains and the uniform distribution over  $[0, 1]^n$ .

All these results are obtained in a unified fashion via simple proofs that are substantially different from previous works [26, 15, 16, 17, 12]. We also give several lower bound examples, including one which shows that our quantitative version of the Gaussian Correlation Inequality is within a logarithmic factor of the best possible bound.

We note that the special case of item 2 above with the uniform distribution on  $\{0, 1\}^n$  essentially recovers Talagrand’s correlation inequality [26]. In more detail, our bound is weaker than that obtained in [26] by a logarithmic factor, but our proof is significantly simpler and easily generalizes to other domains. For  $p$ -biased distributions over  $\{0, 1\}^n$ , our bound avoids any dependence on  $p$  compared to the results of Keller [15, 16, 17] which have a  $p \log(1/p)$  dependence (though, similar to the situation vis-a-vis [26], we lose a logarithmic factor in other dependencies). Finally, for the uniform distribution over  $[0, 1]^n$ , our result gives an answer to a question posed by Keller [16], who wrote “It seems tempting to find a generalization of Talagrand’s result to the continuous setting, but it is not clear what is the correct notion of influences in the continuous case that should be used in such generalization.”

### 1.3 The Approach

We start with a high level meta-observation before explaining our framework and techniques in detail. While the statements of the Harris-Kleitman inequality, the FKG inequality, and the Gaussian Correlation Inequality have a common flavor, the proofs of these results are extremely different from each other. (As noted earlier, the Harris-Kleitman inequality admits a simple inductive proof which is only a few lines long; in contrast the Gaussian Correlation Inequality was an open problem for nearly four decades, and no inductive proof for it is known.) Thus, at first glance, it is not clear how one might come up with a common framework to obtain quantitative versions of these varied qualitative inequalities.

Our approach circumvents this difficulty by using the qualitative inequalities essentially as “black boxes.” This allows us to extend the qualitative inequalities into quantitative ones while essentially sidestepping the difficulties of proving the initial qualitative statements themselves.

#### 1.3.1 Our General Framework

In this subsection we give an overview of our general framework and the high-level ideas underlying it, with our quantitative version of the Gaussian Correlation Inequality serving as a running example throughout for concreteness.

We begin by defining a function  $\Phi : [0, 1] \rightarrow [0, 1]$  which will play an important role in our results:

$$\Phi(x) := \min \left\{ x, \frac{x}{\log^2(1/x)} \right\}. \quad (2)$$

(Note the similarity between  $\Phi$  and the function  $\Psi$  mentioned earlier that arose in Talagrand’s quantitative correlation inequality [26]; the difference is that  $\Phi$  is smaller by essentially a logarithmic factor in the small- $x$  regime.)

Let  $\mathcal{F}$  be a family of real-valued functions on some domain (endowed with measure  $\mu$ ) with  $\mathbf{E}_\mu[f^2] \leq 1$  for all  $f \in \mathcal{F}$ . For example, the Gaussian Correlation Inequality is a correlation inequality for the family  $\mathcal{F}_{\text{csc}}$  of centrally symmetric, convex sets (identified with their 0/1-indicator functions), and  $\mu$  is the standard Gaussian measure  $\mathcal{N}(0, 1)^n$ , usually

denoted  $\gamma$ .<sup>1</sup> A *quantitative* correlation inequality for  $f, g \in \mathcal{F}$  gives a (non-negative) lower bound on the quantity  $\mathbf{E}_{\mathbf{x} \sim \mu}[f(\mathbf{x})g(\mathbf{x})] - \mathbf{E}_{\mathbf{x} \sim \mu}[f(\mathbf{x})] \mathbf{E}_{\mathbf{y} \sim \mu}[g(\mathbf{y})]$ , typically in terms of some measure of “how much  $f$  and  $g$  simultaneously depend on the same coordinates.” Our general approach establishes such a quantitative inequality in two main steps:

**Step 1:** For this step, we require an appropriate family of “noise operators”  $(T_\rho)_{\rho \in [0,1]}$  with respect to the measure  $\mu$ . Very briefly, each of these operators  $T_\rho$  will be a (re-indexed version of a) symmetric Markov semigroup whose stationary distribution is  $\mu$ ; this is defined more precisely in Section 4. (Looking ahead, we will see, for example, that in the case of the GCI, the appropriate noise operator is the Ornstein-Uhlenbeck noise operator, defined in Definition 18.) The crucial property we require of the family  $(T_\rho)_{\rho \in [0,1]}$  with respect to  $\mathcal{F}$  is what we refer to as *monotone compatibility*:

► **Definition 1** (Monotone compatibility). *A class of functions  $\mathcal{F}$  and background measure  $\mu$  is said to be monotone compatible with respect to a family of noise operators  $(T_\rho)_{\rho \in [0,1]}$  if (i) for all  $f, g \in \mathcal{F}$ , the function*

$$q(\rho) := \mathbf{E}_{\mathbf{x} \sim \mu}[f(\mathbf{x})T_\rho g(\mathbf{x})]$$

*is a non-decreasing function of  $\rho$ , and (ii) for  $\rho = 1$  we have  $T_1 = \text{Id}$  (the identity operator).*

The notion of monotone compatibility should be seen as a mild extension of qualitative correlation inequalities. As an example, in the case of the Gaussian Correlation Inequality, Royen’s proof [23] in fact shows that the family  $\mathcal{F}_{\text{csc}}$  is monotone compatible with Ornstein-Uhlenbeck operators.

**Step 2:** We express the operator  $T_\rho$  in terms of its eigenfunctions. In all the cases we consider in this paper, the eigenvalues of the operator  $T_\rho$  are  $\{\rho^j\}_{j \geq 0}$ . Let  $\{\mathcal{W}_j\}_{j \geq 0}$  be the corresponding eigenspaces. Consequently, we can express  $q(\rho) - q(0)$  as

$$q(\rho) - q(0) = \mathbf{E}_{\mathbf{x} \sim \mu}[f(\mathbf{x})T_\rho g(\mathbf{x})] - \mathbf{E}_{\mathbf{x} \sim \mu}[f(\mathbf{x})] \cdot \mathbf{E}_{\mathbf{y} \sim \mu}[g(\mathbf{y})] = \sum_{j > 0} \rho^j \mathbf{E}[f_j(\mathbf{x})g_j(\mathbf{x})], \quad (3)$$

where  $f_j$  (respectively  $g_j$ ) is the projection of  $f$  (respectively  $g$ ) on the space  $\mathcal{W}_j$ . To go back to our running example, for the Gaussian Correlation Inequality,  $\mathcal{W}_j$  is the subspace spanned by degree- $j$  Hermite polynomials on  $\mathbb{R}^n$ .

Define  $a_j := \mathbf{E}[f_j(\mathbf{x})g_j(\mathbf{x})]$ , so  $q(\rho) = \sum_{j \geq 0} a_j \rho^j$ . Now, corresponding to any family  $\mathcal{F}$  and noise operators  $(T_\rho)_{\rho \in [0,1]}$ , there will be a unique  $j^* \in \mathbb{N}$  such that the following properties hold:

1. If  $a_{j^*} = 0$ , then  $\mathbf{E}_{\mathbf{x} \sim \mu}[f(\mathbf{x})g(\mathbf{x})] = \mathbf{E}_{\mathbf{x} \sim \mu}[f(\mathbf{x})] \cdot \mathbf{E}_{\mathbf{y} \sim \mu}[g(\mathbf{y})]$ . In other words,  $a_{j^*}$  *qualitatively* captures the “slack” in the correlation inequality (in fact, as we will soon see,  $a_{j^*}$  also gives a quantitative lower bound on this slack). For example, for the Gaussian Correlation Inequality, it turns out that  $j^* = 2$  (for most of the other applications of our general framework in this paper, it turns out that  $j^* = 1$ ).
2. For any  $i$  such that  $j^*$  does not divide  $i$ ,  $a_i = 0$ .

<sup>1</sup> Since convexity is preserved under linear transformation, no loss of generality is incurred in assuming that the background measure is the standard normal distribution  $\mathcal{N}(0, 1)^n$  rather than an arbitrary centered Gaussian.

Now, from the fact that the spaces  $\{\mathcal{W}_j\}$  are orthonormal and the fact that every  $f \in \mathcal{F}$  has  $\mathbf{E}_\mu[f^2] \leq 1$ , it follows that  $\sum_{j>0} |a_j| \leq 1$ . Our main technical lemma, Lemma 12, implies (see the proof of Theorem 14) that for any such power series  $q(\cdot)$ , there exists some  $\rho^* \in [0, 1]$  such that

$$q(\rho^*) - q(0) \geq \frac{1}{C} \cdot \Phi(a_{j^*}).$$

The proof crucially uses tools from complex analysis. As the class  $\mathcal{F}$  is monotone compatible with the operators  $(T_\rho)_{\rho \in [0,1]}$ , recalling Equation (3), it follows that

$$q(1) - q(0) = \mathbf{E}_{\mathbf{x} \sim \mu} [f(\mathbf{x})g(\mathbf{x})] - \mathbf{E}_{\mathbf{x} \sim \mu} [f(\mathbf{x})] \cdot \mathbf{E}_{\mathbf{y} \sim \mu} [g(\mathbf{y})] \geq \frac{1}{C} \cdot \Phi(a_{j^*}), \quad (4)$$

which is the desired quantitative correlation inequality for  $\mathcal{F}$ .

► **Remark 2.** We emphasize the generality of our framework; the argument sketched above can be carried out in a range of different concrete settings. For example, by using the Harris-Kleitman qualitative correlation inequality for monotone Boolean functions in place of the GCI, and the Bonami-Beckner noise operator over  $\{0, 1\}^n$  in place of the Ornstein-Uhlenbeck noise operator, the above arguments give a simple proof of the following (slightly weaker) version of Talagrand’s correlation inequality (Equation (1)):

$$\mathbf{E}[fg] - \mathbf{E}[f] \mathbf{E}[g] \geq \frac{1}{C} \cdot \Phi \left( \sum_{i=1}^n \mathbf{Inf}_i[f] \mathbf{Inf}_i[g] \right), \quad (5)$$

for an absolute constant  $C > 0$ . While our bound is weaker than that of [26] by a log factor (recall the difference between  $\Psi$  and  $\Phi$ ), our methods are applicable to a much wider range of settings (such as the GCI and the other applications given in this paper). Finally, we emphasize that our proof strategy is really quite different from that of [26]; for example, [26]’s proof relies crucially on bounding the degree-2 Fourier weight of monotone Boolean functions by the degree-1 Fourier weight, whereas our strategy does not analyze the degree-2 spectrum of monotone Boolean functions at all.

► **Remark 3.** Coupled with the first property described above, Equation (4) shows that  $a_{j^*}$  not only qualitatively captures the “correlation gap”

$$\mathbf{E}_{\mathbf{x} \sim \mu} [f(\mathbf{x})g(\mathbf{x})] - \mathbf{E}_{\mathbf{x} \sim \mu} [f(\mathbf{x})] \cdot \mathbf{E}_{\mathbf{y} \sim \mu} [g(\mathbf{y})],$$

but also provides a quantitative lower bound on this gap.

## 2 Preliminaries

In this section we give preliminaries setting notation, recalling useful background on noise operators and orthogonal decomposition of functions over product spaces, and recalling a well-known result that we will require from complex analysis.

### 2.1 Noise Operators and Orthogonal Decompositions

Let  $(\Omega, \pi)$  be a probability space; we do not require  $\Omega$  to be finite, and we assume without loss of generality that  $\pi$  has full support.

The background we require for noise operators on functions in  $L^2(\Omega, \pi)$  is most naturally given using the language of “Markov semigroups.” Our exposition below will be self-contained; for a general and extensive resource on Markov semigroups, we refer the interested reader to [1].



► **Definition 4** (Markov semigroup). *A collection of linear operators  $(P_t)_{t \geq 0}$  on  $L^2(\Omega, \pi)$  is said to be a Markov semigroup if*

1.  $P_0 = \text{Id}$ ;
2. *for all  $s, t \in [0, \infty)$ , we have  $P_s \circ P_t = P_{s+t}$ ; and*
3. *for all  $t \in [0, \infty)$  and all  $f, g \in L^2(\Omega, \pi)$ , the following hold:*
  - a. Identity:  $P_t 1 = 1$  where 1 is the identically-1 function.
  - b. Positivity:  $P_t f \geq 0$  almost everywhere if  $f \geq 0$  almost everywhere.<sup>2</sup>

It is well known that a Markov semigroup can be constructed from a Markov process and vice versa [1]. We call a Markov semigroup *symmetric* if the underlying Markov process is time-reversible; the following definition is an alternative elementary characterization of symmetric Markov semigroups. (Recall that for  $f, g \in L^2(\Omega, \pi)$  the inner product  $\langle f, g \rangle$  is defined as  $\mathbf{E}_{\mathbf{x} \sim \pi}[f(\mathbf{x})g(\mathbf{x})]$ .)

► **Definition 5** (Symmetric Markov semigroup). *A Markov semigroup  $(P_t)_{t \geq 0}$  on  $L^2(\Omega, \pi)$  is symmetric if for all  $t \in [0, \infty)$ , the operator  $P_t$  is self-adjoint; equivalently, for all  $t \in [0, \infty)$  and all  $f, g \in L^2(\Omega, \pi)$ , we have  $\langle f, P_t g \rangle = \langle P_t f, g \rangle$ .*

We note that the families of noise operators  $(U_\rho)_{\rho \in [0,1]}$  and  $(T_\rho)_{\rho \in [0,1]}$  that we consider in Section 5 and Section 6 respectively will be parametrized by  $\rho \in [0, 1]$  where  $\rho = e^{-t}$  for  $t \in [0, \infty)$ , as is standard in theoretical computer science. (For example, the Bonami-Beckner noise operator  $T_\rho$  mentioned in the Introduction, which is a special case of the  $T_\rho$  operator defined in Section 6, corresponds to  $P_t$  for  $(P_t)_{t \geq 0}$  a suitable Markov semigroup and  $\rho = e^{-t}$ .)

Given a Markov semigroup  $(P_t)_{t \geq 0}$  on the probability space  $(\Omega, \pi)$ , we can naturally define the Markov semigroup  $(\otimes_{i=1}^n P_{t_i})_{t_i \geq 0}$  on  $L^2(\Omega^n, \pi^{\otimes n})$ . We write  $P_{\vec{t}}$  to denote this semigroup, and write  $P_t$  to denote the Markov semigroup  $(\otimes_{i=1}^n P_t)_{t \geq 0}$ . We next define a decomposition of  $L^2(\Omega^n, \pi^{\otimes n})$  that is particularly well-suited to the action of a Markov semigroup  $(P_t)_{t \geq 0}$ .

► **Definition 6** (Chaos decomposition). *Consider a Markov semigroup  $(P_t)_{t \geq 0}$  on  $L^2(\Omega^n, \pi^{\otimes n})$ . We call an orthogonal decomposition of*

$$L^2(\Omega^n, \pi^{\otimes n}) = \bigoplus_{i=0}^{\infty} \mathcal{W}_i$$

*a chaos decomposition with respect to the Markov semigroup  $(P_t)_{t \geq 0}$  if*

1.  $\mathcal{W}_0 = \text{span}(1)$  where 1 is the identically-1 function (i.e.  $\mathcal{W}_0 = \mathbb{R}$ ).
2. For all  $t \geq 0$ , there exists  $\lambda_t \in [0, 1]$  such that if  $f \in \mathcal{W}_i$ , then  $P_t f = \lambda_t^i f$ .
3. If  $t_1 > t_2$ , then  $\lambda_{t_1} < \lambda_{t_2}$ .

► **Remark 7.** The term “chaos decomposition” is used in the literature to describe the spectral decomposition of  $L^2(\mathbb{R}^n, \gamma)$  with respect to the Laplacian of the Ornstein–Uhlenbeck semigroup (see Proposition 19); its usage in the broader sense defined above is not standard (to our knowledge).

► **Remark 8.** Given an orthogonal decomposition  $L^2(\Omega^n, \pi^{\otimes n}) = \bigoplus_i \mathcal{W}_i$ , for  $f \in L^2(\Omega^n, \pi^{\otimes n})$  we will write  $f = \bigoplus_i f_i$  where  $f_i$  is the projection of  $f$  onto  $\mathcal{W}_i$ .

<sup>2</sup> Note that this implies the following *order* property: if  $f \geq g$  almost everywhere, then  $P_t f \geq P_t g$  almost everywhere.

We note that  $\lambda_0 = 1$ , and as  $1 \in \mathcal{W}_0$ , it follows that  $f_0 = \langle f, 1 \rangle$ . We revisit the definition of monotone compatibility given in the introduction in the language of Markov semigroups:

► **Definition 9** (Monotone compatibility). *Let  $(P_t)_{t \geq 0}$  be a Markov semigroup on  $L^2(\Omega^n, \pi^{\otimes n})$ . We say that  $(P_t)_{t \geq 0}$  is monotone compatible with a family of functions  $\mathcal{F} \subseteq L^2(\Omega^n, \pi^{\otimes n})$  if for all  $f, g \in \mathcal{F}$ , we have*

$$\frac{\partial}{\partial t} \langle P_t f, g \rangle \leq 0.$$

Recalling that our noise operators such as  $(T_\rho)_{\rho \in [0,1]}$  are reparameterized versions of the Markov semigroup operators  $(P_t)_{t \geq 0}$  under the reparameterization  $T_\rho = P_t$  with  $\rho = e^{-t}$ , and recalling item 1 in Definition 4, we see that Definition 9 coincides with Definition 1.

► **Example 10.** To provide intuition for Definition 9, a useful concrete example to consider is

- $\Omega = \{0, 1\}$  and  $\pi$  = the uniform distribution on  $\Omega$ , so  $L^2(\Omega^n, \pi^{\otimes n})$  is the space of all real-valued functions on the Boolean cube  $\{0, 1\}^n$  under the uniform distribution;
- $\mathcal{F}_{\text{mon}}$  = the class of all monotone Boolean functions, i.e. all  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  such that if  $x_i \leq y_i$  for all  $i$  then  $f(x) \leq f(y)$ ;
- $(P_t)_{t \geq 0}$  is defined by  $P_t = T_{e^{-t}}$ , where  $T_\rho$  is the Bonami-Beckner operator. We remind the reader that for any  $f : \{0, 1\}^n \rightarrow \mathbb{R}$  and any  $0 \leq \rho \leq 1$ , the function  $T_\rho f(x)$  is defined to be  $\mathbf{E}_{\mathbf{y} \sim N_\rho(x)}[f(\mathbf{y})]$ , where “ $\mathbf{y} \sim N_\rho(x)$ ” means that  $\mathbf{y} \in \{0, 1\}^n$  is randomly chosen by independently setting each  $\mathbf{y}_i$  to be  $x_i$  with probability  $\rho$  and to be uniform random with probability  $1 - \rho$ .

In this setting, as will be shown later, we have that for any two monotone functions  $f, g \in \mathcal{F}$ , the function  $\mathbf{E}_{\mathbf{x} \sim \{\pm 1\}^n}[T_\rho f(\mathbf{x})g(\mathbf{x})]$  is a non-decreasing function of  $\rho$ ; hence  $\frac{\partial}{\partial t} \mathbf{E}_{\mathbf{x} \sim \{\pm 1\}^n}[P_t f(\mathbf{x})g(\mathbf{x})]$  is always at most 0 (note that as  $t$  increases  $\rho = e^{-t}$  decreases), so  $(P_t)_{t \geq 0}$  is monotone compatible with  $\mathcal{F}_{\text{mon}}$ .

## 2.2 Complex Analysis

Let  $U \subseteq \mathbb{C}$  be a connected, open set. Recall that a function  $f : U \rightarrow \mathbb{C}$  is said to be *holomorphic* if at every point in  $U$  it is complex differentiable in a neighborhood of the point. For  $U$  a connected closed set,  $f$  is said to be holomorphic if it is holomorphic in an open set containing  $U$ . Our main technical lemma appeals to the following classical result, a proof of which can be found in [24].

► **Theorem 11** (Hadamard Three Circles Theorem). *Suppose  $f$  is holomorphic on the annulus  $\{z \in \mathbb{C} \mid r_1 \leq |z| \leq r_2\}$ . For  $r \in [r_1, r_2]$ , let  $M(r) := \max_{|z|=r} |f(z)|$ . Then*

$$\log \left( \frac{r_2}{r_1} \right) \log M(r) \leq \log \left( \frac{r_2}{r} \right) \log M(r_1) + \log \left( \frac{r}{r_1} \right) \log M(r_2).$$

## 3 A New Extremal Bound for Power Series with Bounded Length

Given a complex power series  $p(t) = \sum_{i=1}^{\infty} c_i t^i$  where  $c_i \in \mathbb{C}$ , its *length* is defined to be the sum of the absolute values of its coefficients, i.e.  $\sum_{i=1}^{\infty} |c_i|$ . Our main technical lemma is a lower bound on the sup-norm of complex power series with no constant term and bounded length:<sup>3</sup>

<sup>3</sup> The “3/2” in the lemma below could be replaced by any constant bounded above 1; we use 3/2 because it is convenient in our later application of Lemma 12.

► **Lemma 12** (Main technical lemma). *Let  $p(t) = \sum_{i=1}^{\infty} c_i t^i$  with  $c_1 = 1$  and  $\sum_{i=1}^{\infty} |c_i| \leq M$  where  $M \geq 3/2$ . Then:*

$$\sup_{t \in [0,1]} |p(t)| \geq \frac{\Theta(1)}{\log^2 M}.$$

The proof given below is inspired by arguments with a similar flavor in [3, 4], where the Hadamard Three Circles Theorem is used to prove various extremal bounds on polynomials.

**Proof.** Consider the meromorphic map (easily seen to have a single pole at  $z = 0$ ) given by

$$h(z) = A \left( z + \frac{1}{z} \right) + B,$$

which maps origin-centered circles to ellipses centered at  $B$ . Let  $0 < \delta < c$  be a parameter that we will fix later, where  $0 < c < 1$  is an absolute constant that will be specified later. We impose the following constraints on  $A$  and  $B$ :

$$-2A + B = \delta \quad \frac{17}{4}A + B = 1,$$

and note that these constraints imply that  $A = \frac{4(1-\delta)}{25}$  and  $B = \frac{8+17\delta}{25}$ .

We define three circles in the complex plane that we will use for the Hadamard Three Circles Theorem:

1. Let  $C_1$  be the circle centered at 0 with radius 1. Note that for all  $z \in C_1$ , the value  $h(z)$  is a real number in the interval  $[\delta, \frac{16+9\delta}{25}] \subseteq [\delta, 1]$ .
2. Let  $r > 1$  be such that  $h(-r) = 0$ , so  $r + \frac{1}{r} = \frac{8+17\delta}{4-4\delta} = 2 + \Theta(\delta)$  and hence  $r = 1 + \Theta(\sqrt{\delta})$ , which is less than 4. Define  $C_2$  to be the circle centered at 0 with radius  $r$ .
3. Let  $C_3$  be the circle centered at 0 with radius 4. Note that  $|h(z)| \leq 1$  for  $z \in C_3$ .

Define  $q(t) := \frac{p(t)}{t}$ . Note that  $q(0) = c_1 = 1$  and that for all  $z \in \mathbb{C}$  such that  $|z| \leq 1$ , we have  $|q(z)| \leq M$ . Define  $\psi(z) := q(h(z))$ . Note that  $\psi$  is holomorphic on  $\mathbb{C} \setminus \{0\}$ ; in particular, it is holomorphic on the annulus defined by  $C_1$  and  $C_3$ . Consequently, by Theorem 11, we have:

$$\log \left( \frac{4}{1} \right) \log \alpha(r) \leq \log \left( \frac{4}{r} \right) \log \alpha(1) + \log \left( \frac{r}{1} \right) \log \alpha(4)$$

with  $\alpha(r) := \sup_{|z|=r} |\psi(z)|$ . As  $h(-r) = 0$ , we have  $\psi(-r) = 1$  and so  $\log \alpha(r) \geq 0$ . Consequently, the left hand side of the above inequality is non-negative, which implies:

$$1 \leq \alpha(1)^{\log \left( \frac{4}{r} \right)} \cdot \alpha(4)^{\log r}.$$

As  $\log \left( \frac{4}{r} \right) = \Theta(1)$ ,  $\log r = \log \left( 1 + \Theta(\sqrt{\delta}) \right) = \Theta(\sqrt{\delta})$ , and  $\alpha(4) \leq M$ , we get:

$$1 \leq \alpha(1)^{\Theta(1)} \cdot M^{\Theta(\sqrt{\delta})}, \quad \text{and hence} \quad M^{-\Theta(\sqrt{\delta})} \leq \alpha(1).$$

By (i) and the definition of  $\alpha$ , we have:

$$\sup_{t \in [\delta, 1]} q(t) \geq M^{-\Theta(\sqrt{\delta})} \quad \text{and hence} \quad \sup_{t \in [0, 1]} p(t) \geq \sup_{\delta \in [0, 1]} \delta M^{-\Theta(\sqrt{\delta})}.$$

Setting  $\delta = \frac{\Theta(1)}{\log^2 M}$ , we get that

$$\sup_{t \in [0, 1]} |p(t)| \geq \frac{\Theta(1)}{\log^2 M},$$

and the lemma is proved. ◀

It is natural to wonder whether Lemma 12 is quantitatively tight. The polynomial  $p(t) = t(1-t)^{\log M}$  is easily seen to have length  $M$  and  $\sup_{t \in [0,1]} p(t) = \Theta(1/\log M)$ , and it is tempting to wonder whether this might be the smallest achievable value. However, it turns out that the  $1/\log^2 M$  dependence of Lemma 12 is in fact the best possible result; a proof of the following result can be found in the appendix to the full version of this paper.

▷ **Claim 13.** For sufficiently large  $M$ , there exists a real polynomial  $p(t) = \sum_{i=1}^d c_i t^i$  with  $c_1 = 1$  and  $\sum_{i=1}^d |c_i| \leq M$  such that

$$\sup_{t \in [0,1]} p(t) \leq O\left(\left(\frac{1}{\log M}\right)^2\right).$$

#### 4 A General Approach to Quantitative Correlation Inequalities

This section presents our general approach to obtaining *quantitative* correlation inequalities from *qualitative* correlation inequalities. While our main result, Theorem 14, is stated in an abstract setting, subsequent sections will instantiate this result in concrete settings that provided the initial impetus for this work. Section 5 deals with the setting of centrally symmetric, convex sets over Gaussian space, and Section 6 deals with finite product domains.

► **Theorem 14 (Main Theorem).** *Consider a symmetric Markov semigroup  $(P_t)_{t \geq 0}$  on  $L^2(\Omega^n, \Pi^{\otimes n})$  with a chaos decomposition*

$$L^2(\Omega^n, \Pi^{\otimes n}) = \bigoplus_{\ell} \mathcal{W}_{\ell}.$$

*Let  $(P_t)_{t \geq 0}$  be monotone compatible with  $\mathcal{F} \subseteq L^2(\Omega^n, \Pi^{\otimes n})$ , where  $\|f\| \leq 1$  for all  $f \in \mathcal{F}$ . Furthermore, suppose that there exists  $j^* \in \mathbb{N}_{>0}$  such that every  $f \in \mathcal{F}$  has a decomposition as*

$$f = \bigoplus_{\ell=0}^{\infty} f_{\ell \cdot j^*},$$

*i.e.  $f_{\ell} = 0$  for  $j^* \nmid \ell$ . Then for all  $f, g \in \mathcal{F}$ , we have*

$$\langle f, g \rangle - f_0 g_0 \geq \frac{1}{C} \cdot \Phi(\langle f_{j^*}, g_{j^*} \rangle), \quad (6)$$

*where recall from Equation (2) that  $\Phi : [0, 1] \rightarrow [0, 1]$  is  $\Phi(x) = \min\left\{x, \frac{x}{\log^2(1/x)}\right\}$  and  $C > 0$  is a universal constant.*

The proof of the above theorem uses an interpolating argument along the Markov semigroup, and appeals to Lemma 12 to obtain the lower bound.

**Proof of Theorem 14.** Fix  $f, g \in \mathcal{F}$  and let us write  $a_{\ell} := \langle f_{\ell}, g_{\ell} \rangle$ . It follows from Definition 6 that  $f_{\ell}, g_{\ell}$  are eigenfunctions of  $P_t$  with eigenvalue  $\lambda_t^{\ell}$ . This, together with the assumption that  $f = \bigoplus_{j^* \mid \ell} f_{\ell}$  and  $g = \bigoplus_{j^* \mid \ell} g_{\ell}$ , implies that for  $t > 0$  we have

$$\langle P_t f, g \rangle = \sum_{j^* \mid \ell} \lambda_t^{\ell} \langle f_{\ell}, g_{\ell} \rangle = \sum_{j^* \mid \ell} a_{\ell} \lambda_t^{\ell}. \quad (7)$$

Here we remark that the argument to  $\Phi(\cdot)$  in the right hand side of Equation (6) is non-negative, i.e.  $a_{j^*} \geq 0$ . To see this, observe that

$$a_{j^*} = \frac{\partial}{\partial \lambda_t^{j^*}} \langle P_t f, g \rangle = \frac{\partial}{\partial t} \langle P_t f, g \rangle \cdot \frac{\partial t}{\partial \lambda_t^{j^*}} \geq 0$$

where we used the monotone compatibility of  $\mathcal{F}$  with  $(P_t)_{t \geq 0}$  and Property 3 of Definition 6.

Returning to Equation (7), rearranging terms gives that

$$\langle P_t f, g \rangle - f_0 g_0 = \sum_{\substack{\ell \geq 0 \\ j^* | \ell}} a_\ell \lambda_t^\ell = a_{j^*} p(\lambda_t^{j^*}), \quad \text{where} \quad p(\lambda_t^{j^*}) := \lambda_t^{j^*} + \frac{1}{a_{j^*}} \sum_{\substack{\ell \geq j^* \\ j^* | \ell}} a_\ell \lambda_t^\ell. \quad (8)$$

As  $\lambda_t \in [0, 1]$ , we re-parametrize  $u := \lambda_t^{j^*}$  and write  $b_\ell := \frac{a_{\ell j^*}}{a_{j^*}}$  for ease of notation; this gives us

$$p(u) = u + \sum_{\ell \geq 2} b_\ell u^\ell.$$

By the Cauchy–Schwarz inequality, we have

$$a_\ell^2 = \langle f_\ell, g_\ell \rangle^2 \leq \langle f_\ell, f_\ell \rangle \langle g_\ell, g_\ell \rangle = \|f_\ell\|^2 \|g_\ell\|^2, \quad \text{and hence} \quad |a_\ell| \leq \|f_\ell\| \|g_\ell\|.$$

Once again using the Cauchy–Schwarz inequality, we get

$$\sum_\ell |a_\ell| \leq \sum_{\ell=0} \|f_\ell\| \cdot \|g_\ell\| \leq \sqrt{\left( \sum_\ell \|f_\ell\|^2 \right) \cdot \left( \sum_\ell \|g_\ell\|^2 \right)} \leq 1$$

where the last inequality follows from the assumption that  $\|f\| \leq 1$  for all  $f \in \mathcal{F}$ . This implies that

$$\sum_\ell |b_\ell| = \frac{1}{|a_{j^*}|} \sum_\ell |a_{\ell j^*}| \leq \frac{1}{|a_{j^*}|} = \frac{1}{a_{j^*}}.$$

where the last equality holds because of  $a_{j^*} \geq 0$  as shown earlier. If  $a_{j^*} > 2/3$  then  $\sum_{\ell \geq 2} |b_\ell| \leq 1/2$  while  $b_1 = 1$ , from which it easily follows that  $\sup_{u \in [0,1]} p(u) \geq 1/2$ . If  $a_{j^*} < 2/3$  then the power series  $p(u)$  satisfies the assumptions of Lemma 12 with  $M = \frac{1}{a_{j^*}}$ . This gives us

$$\sup_{u \in [0,1]} p(u) \geq \min \left\{ \frac{1}{2}, \Theta \left( \frac{1}{\log^2(a_{j^*}^{-1})} \right) \right\}.$$

It follows from Definition 6 that as  $t$  ranges over  $(0, \infty)$ ,  $\lambda_t$  and consequently  $u$  ranges over the interval  $(0, 1]$ . Together with Equation (8), this implies that

$$\sup_{t \in (0, \infty)} \langle P_t f, g \rangle - f_0 g_0 = \sup_{t \in (0, \infty)} a_{j^*} \cdot p(\lambda_t) = a_{j^*} \cdot \sup_{u \in (0, 1]} p(u) \geq \Theta \left( \min \left\{ a_{j^*}, \frac{a_{j^*}}{\log^2(a_{j^*}^{-1})} \right\} \right).$$

However, because of monotone compatibility, we have that  $\langle P_t f, g \rangle$  is decreasing in  $t$ . As  $P_0 = \text{Id}$ , we can conclude that

$$\langle f, g \rangle - f_0 g_0 \geq \Theta \left( \min \left\{ a_{j^*}, \frac{a_{j^*}}{\log^2(a_{j^*}^{-1})} \right\} \right),$$

which completes the proof. ◀

## 5 A Quantitative Extension of the Gaussian Correlation Inequality

In this section, we prove a quantitative versions of Royen’s Gaussian Correlation Inequality (GCI) [23] for symmetric convex sets. We start by recalling some elementary facts about harmonic analysis over Gaussian space in Section 5.1, after which we derive our “robust” form of the Gaussian Correlation Inequality in Section 5.2 as a consequence of Theorem 14. In Section 5.3 we discuss how our robust GCI can be viewed as a Gaussian-space analogue of Talagrand’s celebrated correlation inequality for monotone Boolean functions over the Boolean hypercube [26]. We analyze the tightness of our robust GCI in Section 5.4.

### 5.1 Harmonic (Hermite) Analysis over Gaussian space

Our notation and terminology presented in this subsection follows Chapter 11 of [21]. We say that an  $n$ -dimensional *multi-index* is a tuple  $\alpha \in \mathbb{N}^n$ , and we define

$$\text{supp}(\alpha) := \{i : \alpha_i \neq 0\}, \quad \#\alpha := |\text{supp}(\alpha)|, \quad |\alpha| := \sum_{i=1}^n \alpha_i. \quad (9)$$

We write  $\mathcal{N}(0,1)^n$  to denote the  $n$ -dimensional standard Gaussian distribution. For  $n \in \mathbb{N}_{>0}$ , we write  $L^2(\mathbb{R}^n, \gamma)$  to denote the space of functions  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  that have finite 2<sup>nd</sup> moment  $\|f\|_2^2$  under the standard Gaussian measure  $\gamma$ , that is:

$$\|f\|_2^2 = \mathbf{E}_{\mathbf{z} \sim \mathcal{N}(0,1)^n} [f(\mathbf{z})^2]^{1/2} < \infty.$$

We view  $L^2(\mathbb{R}^n, \gamma)$  as an inner product space with  $\langle f, g \rangle := \mathbf{E}_{\mathbf{z} \sim \mathcal{N}(0,1)^n} [f(\mathbf{z})g(\mathbf{z})]$  for  $f, g \in L^2(\mathbb{R}^n, \gamma)$ . We recall the “Hermite basis” for  $L^2(\mathbb{R}, \gamma)$ :

► **Definition 15** (Hermite basis). *The Hermite polynomials  $(h_j)_{j \in \mathbb{N}}$  are the univariate polynomials defined as*

$$h_j(x) = \frac{(-1)^j}{\sqrt{j!}} \exp\left(\frac{x^2}{2}\right) \cdot \frac{d^j}{dx^j} \exp\left(-\frac{x^2}{2}\right).$$

► **Proposition 16** (Proposition 11.33, [21]). *The Hermite polynomials  $(h_j)_{j \in \mathbb{N}}$  form a complete, orthonormal basis for  $L^2(\mathbb{R}, \gamma)$ . For  $n > 1$  the collection of  $n$ -variate polynomials given by  $(h_\alpha)_{\alpha \in \mathbb{N}^n}$  where*

$$h_\alpha(x) := \prod_{i=1}^n h_{\alpha_i}(x)$$

*forms a complete, orthonormal basis for  $L^2(\mathbb{R}^n, \gamma)$ .*

Given a function  $f \in L^2(\mathbb{R}^n, \gamma)$  and  $\alpha \in \mathbb{N}^n$ , we define its *Hermite coefficient on  $\alpha$*  as  $\tilde{f}(\alpha) = \langle f, h_\alpha \rangle$ . It follows that  $f$  is uniquely expressible as  $f = \sum_{\alpha \in \mathbb{N}^n} \tilde{f}(\alpha) h_\alpha$  with the equality holding in  $L^2(\mathbb{R}^n, \gamma)$ ; we will refer to this expansion as the *Hermite expansion* of  $f$ . One can check that Parseval’s and Plancharel’s identities hold in this setting.

► **Proposition 17** (Plancharel’s identity). *For  $f, g \in L^2(\mathbb{R}^n, \gamma)$ , we have:*

$$\langle f, g \rangle = \mathbf{E}_{\mathbf{z} \sim \mathcal{N}(0,1)^n} [f(\mathbf{z})g(\mathbf{z})] = \sum_{\alpha \in \mathbb{N}^n} \tilde{f}(\alpha) \tilde{g}(\alpha),$$

*and as a special case we have Parseval’s identity,*

$$\langle f, f \rangle = \mathbf{E}_{\mathbf{z} \sim \mathcal{N}(0,1)^n} [f(\mathbf{z})^2] = \sum_{\alpha \in \mathbb{N}^n} \tilde{f}(\alpha)^2.$$

Next we recall the standard Gaussian noise operator (parameterized so that the noise rate  $\rho$  ranges over  $[0, 1]$ ):

► **Definition 18** (Ornstein-Uhlenbeck semigroup). *We define the Ornstein-Uhlenbeck semigroup as the family of operators  $(U_\rho)_{\rho \in [0,1]}$  on the space of functions  $f \in L^1(\mathbb{R}^n, \gamma)$  given by*

$$U_\rho f(x) := \mathbf{E}_{\mathbf{g} \sim \mathcal{N}(0,1)^n} \left[ f\left(\rho \cdot x + \sqrt{1-\rho} \cdot \mathbf{g}\right) \right].$$

The Ornstein-Uhlenbeck semigroup is sometimes referred to as the family of *Gaussian noise operators* or *Mehler transforms*. The Ornstein-Uhlenbeck semigroup acts on the Hermite expansion as follows:

► **Proposition 19** (Proposition 11.33, [21]). *For  $f \in L^2(\mathbb{R}^n, \gamma)$ , the function  $U_\rho f$  has Hermite expansion*

$$U_\rho f = \sum_{\alpha \in \mathbb{N}^n} \rho^{|\alpha|} \tilde{f}(\alpha) h_\alpha.$$

## 5.2 A Robust Extension of the Gaussian Correlation Inequality

We start by making a crucial observation regarding Royen's proof of the Gaussian correlation inequality (GCI) [23]. Recall that the GCI states that if  $K$  and  $L$  are the indicator functions of two centrally symmetric (i.e.  $K(x) = 1$  implies  $K(-x) = 1$ ), convex sets, then they are non-negatively correlated under the Gaussian measure; that is,

$$\mathbf{E}_{\mathbf{x} \sim \mathcal{N}(0,1)^n} [K(\mathbf{x})L(\mathbf{x})] - \mathbf{E}_{\mathbf{x} \sim \mathcal{N}(0,1)^n} [K(\mathbf{x})] \mathbf{E}_{\mathbf{y} \sim \mathcal{N}(0,1)^n} [K(\mathbf{y})] \geq 0.$$

In order to prove this, Royen interpolates between  $\mathbf{E}[K] \mathbf{E}[L]$  and  $\mathbf{E}[KL]$  via the Ornstein-Uhlenbeck semigroup, and shows that this interpolation is monotone nondecreasing; indeed, note that

$$\langle U_1 K, L \rangle = \mathbf{E}_{\mathbf{x} \sim \mathcal{N}(0,1)^n} [K(\mathbf{x})L(\mathbf{x})], \quad \text{and that} \quad \langle U_0 K, L \rangle = \mathbf{E}_{\mathbf{x} \sim \mathcal{N}(0,1)^n} [K(\mathbf{x})] \mathbf{E}_{\mathbf{y} \sim \mathcal{N}(0,1)^n} [K(\mathbf{y})].$$

Thus, Royen's main result can be interpreted as follows (we refer the interested reader to a simplified exposition of Royen's proof by Latała and Matlak [20] for further details):

► **Proposition 20** (Royen's Theorem, [23]). *Let  $\mathcal{F}_{\text{csc}} \subseteq L^2(\mathbb{R}^n, \gamma)$  be the family of indicators of centrally symmetric, convex sets, and let  $(U_\rho)_{\rho \in [0,1]}$  be the Ornstein-Uhlenbeck semigroup. Then for  $K, L \in \mathcal{F}_{\text{csc}}$ , we have*

$$\frac{\partial}{\partial \rho} \langle U_\rho K, L \rangle \geq 0 \quad \text{for all } 0 < \rho < 1.$$

In particular,  $\mathcal{F}_{\text{csc}}$  is monotone compatible with  $(U_\rho)_{\rho \in [0,1]}$ .

Recall that we are parametrizing the Ornstein-Uhlenbeck semigroup by  $\rho \in [0, 1]$  where  $\rho = e^{-t}$  for  $t \in [0, \infty)$ ; see the discussion following Definition 4. We can now state our main result:

► **Theorem 21** (Quantitative GCI). *Let  $\mathcal{F}_{\text{csc}} \subseteq L^2(\mathbb{R}^n, \gamma)$  be the family of indicators of centrally symmetric, convex sets. Then for  $K, L \in \mathcal{F}_{\text{csc}}$ , we have*

$$\mathbf{E}[KL] - \mathbf{E}[K] \mathbf{E}[L] \geq \frac{1}{C} \cdot \Phi \left( \sum_{|\alpha|=2} \tilde{K}(\alpha) \tilde{L}(\alpha) \right)$$

where recall from Equation (2) that  $\Phi : [0, 1] \rightarrow [0, 1]$  is  $\Phi(x) = \min \left\{ x, \frac{x}{\log^2(1/x)} \right\}$  and  $C > 0$  is a universal constant.



**Proof.** Consider the orthogonal decomposition

$$L^2(\mathbb{R}^n, \gamma) = \bigoplus_{i=0}^{\infty} \mathcal{W}_i$$

where  $\mathcal{W}_i = \text{span}\{h_\alpha : |\alpha| = i\}$ ; the orthogonality of this decomposition follows from Proposition 16. From Proposition 19, it follows that this decomposition is in fact a chaos decomposition (recall Definition 6) with respect to the Ornstein-Uhlenbeck semigroup  $(U_\rho)_{\rho \in [0,1]}$ .

If  $K \in \mathcal{F}_{\text{csc}}$ , then  $K(x) = K(-x)$  as  $K$  is the indicator of a centrally symmetric set; in other words,  $K$  is an even function. Consequently, its Hermite expansion is given by

$$K = \bigoplus_{\substack{i=0 \\ |\alpha|=2i}}^{\infty} h_\alpha.$$

Furthermore, from Proposition 17, we have that

$$\|K\|^2 = \sum_{\alpha \in \mathbb{N}^n} \tilde{K}(\alpha)^2 = \mathbf{E}[K^2] \leq 1.$$

It follows that the hypotheses of Theorem 14 hold for  $\mathcal{F}_{\text{csc}}$  with  $j^* = 2$ ; consequently, for  $K, L \in \mathcal{F}_{\text{csc}}$  we have

$$\langle U_1 K, L \rangle - \langle U_0 K, L \rangle = \mathbf{E}[KL] - \mathbf{E}[K] \mathbf{E}[L] \geq \frac{1}{C} \cdot \Phi \left( \sum_{|\alpha|=2} \tilde{K}(\alpha) \tilde{L}(\alpha) \right),$$

which completes the proof of the theorem.  $\blacktriangleleft$

It is natural to ask whether Theorem 21 can be extended to a broader class of functions than 0/1-valued indicator functions of centrally symmetric, convex sets  $\mathcal{F}_{\text{csc}}$ . Indeed, the GCI implies the monotone compatibility of centrally symmetric, *quasiconcave*<sup>4</sup>, non-negative functions (which is a larger family of functions than  $\mathcal{F}_{\text{csc}}$ ) with the Ornstein-Uhlenbeck semigroup. This allows us to once again use Theorem 14 to obtain a quantitative correlation inequality for this family of functions; we defer this to the full version of this paper.

### 5.3 Interpreting Theorem 21

Recall Talagrand's correlation inequality [26]: If  $f, g : \{0, 1\}^n \rightarrow \{0, 1\}$  are monotone Boolean functions, then

$$\mathbf{E}[fg] - \mathbf{E}[f] \mathbf{E}[g] \geq \frac{1}{C} \cdot \Psi \left( \sum_{i=1}^n \hat{f}(i) \hat{g}(i) \right)$$

where  $\Psi(x) = \frac{x}{\log(e/x)}$ . However (see Chapter 2 of [21]), for monotone  $f : \{+1, -1\}^n \rightarrow \{+1, -1\}$ , we have  $\hat{f}(i) = \mathbf{Inf}_i[f]$  where

$$\mathbf{Inf}_i[f] := \mathbf{Pr}_{\mathbf{x} \sim \{+1, -1\}^n} [f(\mathbf{x}) \neq f(\mathbf{x}^{\oplus i})].$$

<sup>4</sup> A function  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  is *quasiconcave* if for all  $\lambda \in [0, 1]$  we have  $f(\lambda x + (1 - \lambda)y) \geq \min\{f(x), f(y)\}$ .

In other words, the degree-1 Fourier coefficient  $\widehat{f}(i)$  captures the “dependence” of  $f$  on its  $i^{\text{th}}$  coordinate, and the quantity  $\sum_{i=1}^n \widehat{f}(i)\widehat{g}(i)$  captures the extent to which “both  $f$  and  $g$  simultaneously depend on the same coordinates”. This intuitively explains why it is plausible for such a quantity to appear in Talagrand’s inequality.

Inspired by the resemblance between our quantitative Gaussian correlation inequality and Talagrand’s correlation inequality, we believe that the (negated) degree-2 Hermite coefficients of centrally symmetric, convex sets over Gaussian space are natural analogues of the degree-1 Fourier coefficients (i.e. the coordinate influences) of monotone Boolean functions. However, while functions on the Boolean hypercube have influences only along  $n$  “directions”, there are infinitely many directions over Gaussian space. We make the following definition:

► **Definition 22** (Influences for  $\mathcal{F}_{\text{csc}}$ ). *Let  $K \subseteq \mathbb{R}^n$  be a centrally symmetric, convex set. Given a unit vector  $v \in S^{n-1}$ , we define the influence of  $K$  along direction  $v$  as*

$$\mathbf{Inf}_v[K] := -\widetilde{K}(2v) = \mathbf{E}_{\mathbf{x} \sim \mathcal{N}(0,1)^n} [-K(\mathbf{x})h_2(v \cdot \mathbf{x})]$$

where  $h_2(x) = \frac{x^2-1}{\sqrt{2}}$  is the degree-2 univariate Hermite polynomial (see Section 11.2 of [21]).

It follows from the proof of Theorem 14 that the quantity  $\sum_{|\alpha|=2} \widetilde{K}(\alpha)\widetilde{L}(\alpha)$  for  $K, L \in \mathcal{F}_{\text{csc}}$  is non-negative. In fact more is true: if  $K$  is a centrally symmetric, convex set, then each  $\mathbf{Inf}_{e_i}[K]$  is itself non-negative. The proof of the following proposition can be found in the appendix to the full version of this paper.

► **Proposition 23** (Influences are non-negative). *If  $K$  is a centrally symmetric, convex set, then  $\mathbf{Inf}_v[K] \geq 0$  for all  $v \in S^{n-1}$ , with equality holding if and only if  $K(x) = K(y)$  whenever  $x_{v^\perp} = y_{v^\perp}$  (the projection of  $x$  orthogonal to  $v$  coincides with that of  $y$ ), except possibly on a set of measure zero.*

It is natural to define the “total influence of  $K$ ” to be  $\mathbf{Inf}[K] := \sum_{i=1}^n \mathbf{Inf}_{e_i}[K]$ ; we observe that this quantity is given by

$$\mathbf{Inf}[K] = -\sum_{i=1}^n \widetilde{K}(2e_i) = \frac{\mathbf{E}_{\mathbf{x} \sim \mathcal{N}(0,1)^n} [-f(\mathbf{x}) \cdot (\|\mathbf{x}\|^2 - n)]}{\sqrt{2}},$$

and hence it is invariant under orthogonal transformations (i.e. any orthonormal basis  $v_1, \dots, v_n$  could have been used in place of  $e_1, \dots, e_n$  in defining  $\mathbf{Inf}[K]$ ).

The above discussion suggests that the notion of “influences” for centrally symmetric, convex sets in Gaussian space proposed in Definition 22 is indeed “influence-like”. A forthcoming paper [5] will further explore this notion.

## 5.4 On the Tightness of Theorem 21

In [26], Talagrand gave the following family of example functions for which Equation (1) is tight up to constant factors: let  $f, g : \{0, 1\}^n \rightarrow \{0, 1\}$  be given by

$$f(x) = \begin{cases} 1 & \sum_i x_i \geq n - k \\ 0 & \text{otherwise} \end{cases}, \quad \text{and} \quad g(x) = \begin{cases} 1 & \sum_i x_i > k \\ 0 & \text{otherwise} \end{cases}$$

where  $k \leq n/2$ . Writing  $\epsilon$  to denote  $\mathbf{E}[f]$ , we have  $\epsilon^2 = \epsilon - \epsilon(1 - \epsilon) = \mathbf{E}[fg] - \mathbf{E}[f]\mathbf{E}[g]$ , and it can be shown that  $\Psi\left(\sum_{i=1}^n \widehat{f}(i)\widehat{g}(i)\right) = \Theta(\epsilon^2)$ , so Equation (1) is tight up to constant factors. We note that in this example  $f$  and  $g$  are the indicator functions of Hamming balls, and that  $f \subseteq g$  (i.e.  $f(x) = 1$  implies that  $g(x) = 1$ ).

Motivated by this example, we consider an analogous pair of functions in the setting of centrally symmetric, convex sets over Gaussian space, where we use origin-centered balls of different radii in place of Hamming balls. In particular, let  $K, L \in \mathcal{F}_{\text{csc}}$  be  $n$ -dimensional origin-centered balls of radii  $r_1$  and  $r_2$  respectively such that  $r_1 < r_2$  with

$$\mathbf{E}[K] = \epsilon \quad \text{and} \quad \mathbf{E}[L] = 1 - \epsilon$$

where the expectations are taken with respect to the  $n$ -dimensional Gaussian measure. As  $K \subseteq L$ , we have  $\mathbf{E}[KL] - \mathbf{E}[K]\mathbf{E}[L] = \epsilon - \epsilon(1 - \epsilon) = \epsilon^2$ . Since  $K(x_1, \dots, x_n) = K(x_1, \dots, x_{i-1}, -x_i, x_{i+1}, \dots, x_n)$  for all  $x \in \mathbb{R}^n$  and all  $i \in [n]$ , it easily follows that  $\tilde{K}(e_i + e_j) = \mathbf{E}[K(\mathbf{x})\mathbf{x}_i\mathbf{x}_j] = 0$  for all  $i \neq j$ , and the same is true for  $L$ . Furthermore, as  $K, L$  are rotationally invariant, we have  $\tilde{K}(2e_i) = \tilde{K}(2e_j)$  and  $\tilde{L}(2e_i) = \tilde{L}(2e_j)$  for all  $i, j \in [n]$ . It follows that

$$-\sum_{|\alpha|=2} \tilde{K}(\alpha) = -\sum_{i=1}^n \tilde{K}(2e_i) = \frac{1}{\sqrt{2}} \mathbf{E}_{\mathbf{x} \sim \mathcal{N}(0,1)^n} [K(\mathbf{x}) (n - \|\mathbf{x}\|^2)].$$

An application of the Berry-Esseen Central Limit Theorem (see [2, 6] or, for example, Section 11.5 of [21]) together with standard anti-concentration bounds on Gaussian tails (we omit the details here; a complete calculation can be found in the full version of this paper) gives that

$$\mathbf{E}_{\mathbf{x} \sim \mathcal{N}(0,1)^n} [K(\mathbf{x}) (n - \|\mathbf{x}\|^2)] = \Omega\left(\epsilon \sqrt{n \ln\left(\frac{2}{\epsilon}\right)}\right).$$

A similar calculation for  $L$  gives that  $-\tilde{L}(2e_i) = \Omega\left(\epsilon \sqrt{\frac{1}{n} \ln\left(\frac{2}{\epsilon}\right)}\right)$ , from which it follows that  $\sum_{i=1}^n \tilde{K}(2e_i) \tilde{L}(2e_i) = \Omega(\epsilon^2 \ln(\frac{2}{\epsilon}))$ . Recalling Equation (2), we get that for small enough  $\epsilon$ , the quantity

$$\Phi\left(\sum_{|\alpha|=2} \tilde{K}(\alpha) \tilde{L}(\alpha)\right) = \Omega\left(\frac{\epsilon^2}{\log(2/\epsilon)}\right),$$

which lets us conclude that Theorem 21 is tight to within a logarithmic factor.

## 6 Generalizing Talagrand's Inequality to Arbitrary Finite Product Domains

The main result of this section, Theorem 29, is an extension of Talagrand's correlation inequality [26] to real-valued functions on general, finite, product spaces. (Recall that Talagrand's inequality applies only to Boolean-valued functions on the domain  $\{0, 1\}^n$  under the uniform distribution.)

### 6.1 Harmonic Analysis over Finite Product Spaces

Our notation and terminology presented in this subsection follows Chapter 8 of [21]. We use multi-index notation for  $\alpha \in \mathbb{N}^n$  as defined in Equation (9).

Let  $(\Omega, \pi)$  be a finite probability space with  $|\Omega| = m \geq 2$ , where we always assume that the distribution  $\pi$  over  $\Omega$  has full support (i.e.  $\pi(\omega) > 0$  for every  $\omega \in \Omega$ ). We write  $L^2(\Omega^n, \pi^{\otimes n})$  for the real inner product space of functions  $f : \Omega^n \rightarrow \mathbb{R}$ , with inner product  $\langle f, g \rangle := \mathbf{E}_{\mathbf{x} \sim \pi^{\otimes n}} [f(\mathbf{x})g(\mathbf{x})]$ .

It is easy to see that there exists an orthonormal basis for the inner product space  $L^2(\Omega, \pi)$ , i.e. a set of functions  $\phi_0, \dots, \phi_{m-1} : \Omega \rightarrow \mathbb{R}$ , with  $\phi_0 = 1$ , that are orthonormal with respect to  $\pi$ . Moreover, such a basis extends to an orthonormal basis for  $L^2(\Omega^n, \pi^{\otimes n})$  by a straightforward  $n$ -fold product construction: given a multi-index  $\alpha \in \mathbb{N}_{<m}^n$ , if we define  $\phi_\alpha \in L^2(\Omega^n, \pi^{\otimes n})$  as

$$\phi_\alpha(x) := \prod_{i=1}^n \phi_{\alpha_i}(x_i),$$

then the collection  $(\phi_\alpha)_{\alpha \in \mathbb{N}_{<m}^n}$  is an orthonormal basis for  $L^2(\Omega^n, \pi^{\otimes n})$  (see Proposition 8.13 of [21]). So every function  $f : \Omega^n \rightarrow \mathbb{R}$  has a decomposition

$$f = \sum_{\alpha \in \mathbb{N}_{<m}^n} \hat{f}(\alpha) \phi_\alpha. \quad (10)$$

This can be thought of as a “Fourier decomposition” for  $f$ , in that it satisfies both Parseval’s and Plancherel’s identities (see Proposition 8.16 of [21]). We now proceed to define a noise operator for finite product spaces.

► **Definition 24** (Noise operator for finite product spaces). *Fix a finite product probability space  $L^2(\Omega^n, \pi^{\otimes n})$ . For  $\rho \in [0, 1]$  we define the noise operator for  $L^2(\Omega^n, \pi^{\otimes n})$  as the linear operator*

$$T_\rho f(x) := \mathbf{E}_{\mathbf{y} \sim N_\rho(x)}[f(\mathbf{y})],$$

where “ $\mathbf{y} \sim N_\rho(x)$ ” means that  $\mathbf{y} \in \Omega^n$  is randomly chosen as follows: for each  $i \in [n]$ , with probability  $\rho$  set  $\mathbf{y}_i$  to be  $x_i$  and with the remaining  $1 - \rho$  probability set  $\mathbf{y}_i$  by independently making a draw from  $\pi$ .

► **Proposition 25** (Proposition 8.28 of [21]). *We have  $T_\rho f = \sum_{\alpha} \rho^{\#\alpha} \hat{f}(\alpha) \phi_\alpha$ .*

## 6.2 A Quantitative Correlation Inequality for Finite Product Domains

Throughout this subsection, let  $\Omega = \{0, 1, \dots, m-1\}$  endowed with the natural ordering (though any  $m$ -element totally ordered set would do). We will consider monotone functions on  $(\Omega^n, \pi^{\otimes})$ ; while our results hold in the more general setting of functions on  $(\Omega^n, \otimes_{i=1}^n \pi_i)$ , we stick to the setting of  $L^2(\Omega^n, \pi^{\otimes n})$  for ease of exposition.

In order to appeal to Theorem 14, we must first show that the family of monotone (nondecreasing) functions on  $\Omega^n$  is monotone compatible with the Bonami–Beckner noise operator (see Definition 24). To this end, we define noise operators that act on each coordinate of the input:

► **Definition 26** (coordinate-wise noise operators). *Let  $T_\rho^i$  be the operator on functions  $f : \Omega^n \rightarrow \mathbb{R}$  defined by*

$$T_\rho^i f(x) = \mathbf{E}_{\mathbf{y} \sim N_\rho(x_i)}[f(x_1, \dots, \mathbf{y}, \dots, x_n)],$$

and define  $T_{\rho_1, \dots, \rho_n} f := T_{\rho_1}^1 \circ T_{\rho_2}^2 \circ \dots \circ T_{\rho_n}^n f$ .

This is well-defined as the operators  $T_{\rho_i}^i$  and  $T_{\rho_j}^j$  commute.

► **Lemma 27.** *Let  $\Omega = \{0, 1, \dots, m-1\}$  and let  $f : \Omega^n \rightarrow \mathbb{R}$  be a monotone function. Then  $T_\rho^i f : \Omega^n \rightarrow \mathbb{R}$  is a monotone function.*

**Proof.** Suppose  $x, y \in \Omega^n$  are such that  $x_i \leq y_i$  for all  $i \in [n]$ . We wish to show that  $T_\rho^i f(x) \leq T_\rho^i f(y)$ , which is equivalent to showing

$$\mathbf{E}_{z \sim N_\rho(x_i)} [f(x^{i \rightarrow z})] \leq \mathbf{E}_{z \sim N_\rho(y_i)} [f(y^{i \rightarrow z})].$$

Indeed, because of the monotonicity of  $f$ , via the natural coupling we have

$$\begin{aligned} \mathbf{E}_{z \sim N_\rho(x_i)} [f(x^{i \rightarrow z})] &= \delta f(x) + (1 - \delta) \mathbf{E}_{z \sim \Omega^n} [f(x^{i \rightarrow z})] \\ &\leq \delta f(y) + (1 - \delta) \mathbf{E}_{z \sim \Omega^n} [f(y^{i \rightarrow z})] = \mathbf{E}_{z \sim N_\rho(y_i)} [f(y^{i \rightarrow z})]. \quad \blacktriangleleft \end{aligned}$$

► **Lemma 28.** Let  $\Omega = \{0, 1, \dots, m-1\}$  and let  $f, g : \Omega^n \rightarrow \mathbb{R}$  be monotone functions. Then  $\langle T_\rho f, g \rangle$  is nondecreasing in  $\rho \in [0, 1]$ .

**Proof.** We have

$$\langle T_{\rho_1, \dots, \rho_n} f, g \rangle = \langle T_{\rho_1, \dots, 1} f, T_{1, \rho_2, \dots, \rho_n} g \rangle = \langle T_{\rho_1}^1 f, h \rangle$$

where  $h := T_{1, \rho_2, \dots, \rho_n} g$ . It follows from a repeated application of Lemma 27 that  $h$  is monotone. Now, note that

$$\langle T_{\rho_1}^1 f, h \rangle = \hat{f}(\bar{0}) \cdot \hat{h}(\bar{0}) + \sum_{\alpha_1 > 0} \rho_1 \hat{f}(\alpha) \hat{h}(\alpha) + \sum_{\substack{\bar{0} \neq \alpha \\ \alpha_1 = 0}} \hat{f}(\alpha) \hat{h}(\alpha)$$

where  $\bar{0} = (0, \dots, 0)$ . By Cheybshev's order inequality, we know that  $\langle T_1^1 f, h \rangle \geq \langle T_0^1 f, h \rangle = \hat{f}(\bar{0}) \cdot \hat{h}(\bar{0}) + \sum_{\bar{0} \neq \alpha, \alpha_1 = 0} \hat{f}(\alpha) \hat{h}(\alpha)$ . From the above expression, we have:

$$\frac{\partial}{\partial \rho_1} \langle T_{\rho_1}^1 f, h \rangle = \sum_{\alpha_1 > 0} \hat{f}(\alpha) \hat{h}(\alpha)$$

which must be nonnegative since  $\langle T_1^1 f, h \rangle \geq \langle T_0^1 f, h \rangle$ , and so we can conclude that  $\langle T_{\rho_1}^1 f, h \rangle$  is nondecreasing in  $\rho_1$ . The result then follows by repeating this for each coordinate. ◀

Let  $\mathcal{F}_{\text{mon}} \subseteq L^2(\Omega^n, \pi^{\otimes n})$  be the family of monotone functions  $f : \Omega^n \rightarrow \mathbb{R}$ . Then Lemma 28 shows that  $\mathcal{F}_{\text{mon}}$  is monotone compatible with the Bonami–Beckner noise operator. We can now prove our Talagrand-analogue for monotone functions over  $\Omega^n$ :

► **Theorem 29.** Let  $\Omega = \{0, 1, \dots, m-1\}^n$  and let  $\mathcal{F}_{\text{mon}} \subseteq L^2(\Omega^n, \pi^{\otimes n})$  denote the family of monotone functions on  $\Omega^n$  such that  $\|f\| \leq 1$  for all  $f \in \mathcal{F}_{\text{mon}}$ . Then for  $f, g \in \mathcal{F}_{\text{mon}}$ , we have

$$\mathbf{E}[fg] - \mathbf{E}[f] \mathbf{E}[g] \geq \frac{1}{C} \cdot \Phi \left( \sum_{\# \alpha = 1} \hat{f}(\alpha) \hat{g}(\alpha) \right)$$

where recall from Equation (2) that  $\Phi : [0, 1] \rightarrow [0, 1]$  is  $\Phi(x) = \min \left\{ x, \frac{x}{\log^2(1/x)} \right\}$  and  $C > 0$  is a universal constant.

**Proof.** Consider the orthogonal decomposition

$$L^2(\Omega^n, \pi^{\otimes n}) = \bigoplus_{i=0}^n \mathcal{W}_i$$

where  $\mathcal{W}_i = \text{span} \{ \phi_\alpha : \# \alpha = i \}$ ; the orthogonality of this decomposition follows from the orthonormality of  $(\phi_\alpha)_{\alpha \in \mathbb{N}_{< m}^n}$ . Furthermore, this decomposition is a chaos decomposition with respect to the Bonami–Beckner operator  $(T_\rho)_{\rho \in [0, 1]}$ . It follows that the hypotheses of Theorem 14 hold for  $\mathcal{F}_{\text{mon}}$  with  $j^* = 1$ , from which the result follows. ◀

Theorem 29 can be interpreted in terms of the *Efron–Stein decomposition* of a function (see Chapter 8 of [21]); a complete discussion of this can be found in the full version of this paper.

### 6.3 Comparison with Keller’s Inequality for the $p$ -biased Hypercube

In this subsection, we restrict our attention to the  $p$ -biased hypercube, i.e.  $\{+1, -1\}_p^n := (\{+1, -1\}^n, \pi_p^{\otimes n})$  where  $\pi_p(-1) = p$  and  $\pi_p(+1) = 1 - p$ . In this setting our Theorem 29 generalizes Talagrand’s inequality in two ways: it holds for real-valued monotone functions on  $\{+1, -1\}^n$  that have 2-norm at most 1 (rather than just monotone Boolean functions), and it holds for any  $p$  (as opposed to just  $p = 1/2$ ). Keller [15, 16] has earlier given a generalization of Talagrand’s inequality that holds for general  $p$  and for real-valued monotone functions with  $\infty$ -norm at most 1:

► **Theorem 30** (Theorem 7 of [15]; see also [17] for a slightly weaker version). *Let  $f, g \in L^2(\{0, 1\}^n, \pi_p^{\otimes n})$  be monotone functions such that for all  $x \in \{+1, -1\}^n$ , we have  $|f(x)|, |g(x)| \leq 1$ . Then*

$$\mathbf{E}[fg] - \mathbf{E}[f] \mathbf{E}[g] \geq \frac{1}{C} \cdot H(p) \cdot \Psi \left( \sum_{i=1}^n \hat{f}_p(i) \hat{g}_p(i) \right)$$

where  $\hat{f}_p(i)$  is the  $p$ -biased degree-1 Fourier coefficient on coordinate  $i$ ,  $\Psi : [0, 1] \rightarrow [0, 1]$  is given by  $\Psi(x) = \frac{x}{\log(e/x)}$  as in Section 1.1,  $C > 0$  is a universal constant, and  $H : [0, 1] \rightarrow [0, 1]$  is the binary entropy function  $H(x) = -x \log x - (1 - x) \log(1 - x)$ .

Comparing Theorem 29 to Theorem 30, we see that the latter has an extra factor of  $H(p)$ , whereas the former shows that in fact no dependence on  $p$  is necessary (but the former has an extra factor of  $1/\log \left( 1 / \sum_i \hat{f}_p(i) \hat{g}_p(i) \right)$ ). Theorem 29 can be significantly stronger than Theorem 30 in a range of natural settings because of these differences. We show that for every  $\omega(1)/n \leq p \leq 1/2$ , there is a pair of  $\{+1, -1\}$ -valued functions  $f, g$  (depending on  $p$ ) such that under the  $p$ -biased distribution (i) the quantity  $\mathbf{E}[fg] - \mathbf{E}[f] \mathbf{E}[g]$  is at least an absolute constant independent of  $n$  and  $p$ ; (ii) the RHS of Theorem 29 is at least an absolute constant independent of  $n$  and  $p$ ; but (iii) the RHS of Theorem 30 is  $\Theta(p \log(1/p))$ . A proof can be found in the appendix to the full version of this paper.

---

#### References

- 1 D. Bakry, I. Gentil, and M. Ledoux. *Analysis and Geometry of Markov Diffusion Operators*. Springer, 2013. URL: <https://books.google.com/books?id=tf37sgEACAAJ>.
- 2 Andrew C. Berry. The accuracy of the Gaussian approximation to the sum of independent variates. *Transactions of the American Mathematical Society*, 49(1):122–136, 1941.
- 3 Peter Borwein and Tamás Erdélyi. Littlewood-type polynomials on subarcs of the unit circle. *Indiana University Mathematics Journal*, 46(4):1323–1346, 1997.
- 4 Peter Borwein, Tamás Erdélyi, and Géza Kós. Littlewood-type problems on  $[0, 1]$ . *Proceedings of the London Mathematical Society*, 3(79):22–46, 1999.
- 5 A. De, S. Nadimpalli, and R. Servedio. Influences for Centrally Symmetric, Convex Sets. In preparation, 2020.
- 6 Carl-Gustav Esseen. On the Liapunoff limit of error in the theory of probability. *Arkiv för matematik, astronomi och fysik*, A:1–19, 1942.
- 7 C. M. Fortuin, P. W. Kasteleyn, and J. Ginibre. Correlation inequalities on some partially ordered sets. *Comm. Math. Phys.*, 22(2):89–103, 1971. URL: <https://projecteuclid.org:443/euclid.cmp/1103857443>.

- 8 G. Gallavotti. A proof of the Griffiths inequalities for the XY model. *Stud. Appl. Math.*, 50(1):89–92, 1971.
- 9 R. Griffiths. Correlations in Ising ferromagnets. I. *Journal of Mathematical Physics*, 8(3):478–483, 1967.
- 10 T.E. Harris. A lower bound for the critical probability in a certain percolation process. *Proc. Camb. Phil. Soc.*, 56:13–20, 1960.
- 11 Yaozhong Hu. Itô-wiener chaos expansion with exact residual and correlation, variance inequalities. *Journal of Theoretical Probability*, 10(4):835–848, 1997.
- 12 G. Kalai, N. Keller, and E. Mossel. On the correlation of increasing families. *Journal of Combinatorial Theory, Series A*, 144, November 2015. doi:10.1016/j.jcta.2016.06.012.
- 13 N. Keller. Lower bound on the correlation between monotone families in the average case. *Advances in Applied Mathematics*, 43(1):31–45, 2009.
- 14 N. Keller, E. Mossel, and A. Sen. Geometric Influences II: Correlation Inequalities and Noise Sensitivity. *Ann. Inst. H. Poincaré Probab. Statist.*, 50(4):1121–1139, November 2014. doi:10.1214/13-AIHP557.
- 15 Nathan Keller. Improved FKG Inequality for product measures on the discrete cube, 2008.
- 16 Nathan Keller. *Influences of variables on Boolean functions*. PhD thesis, Hebrew University of Jerusalem, 2009.
- 17 Nathan Keller. A simple reduction from a biased measure on the discrete cube to the uniform measure. *European Journal of Combinatorics*, 33:1943–1957, 2012.
- 18 D. Kelly and S. Sherman. General Griffiths’ inequalities on correlations in Ising ferromagnets. *Journal of Mathematical Physics*, 9(3):466–484, 1968.
- 19 Daniel J Kleitman. Families of non-disjoint subsets. *Journal of Combinatorial Theory*, 1(1):153–155, 1966.
- 20 Rafał Łatała and Dariusz Matlak. Royen’s Proof of the Gaussian Correlation Inequality. *Geometric Aspects of Functional Analysis*, pages 265–275, 2017. doi:10.1007/978-3-319-45282-1\_17.
- 21 R. O’Donnell. *Analysis of Boolean functions*. Cambridge University Press, 2014.
- 22 Christopher J. Preston. A generalization of the FKG inequalities. *Communications in Mathematical Physics*, 36(3):233–241, 1974.
- 23 Thomas Royen. A simple proof of the Gaussian correlation conjecture extended to multivariate gamma distributions. *arXiv preprint*, 2014. arXiv:1408.1028.
- 24 Walter Rudin. *Real and Complex Analysis, 3rd Ed.* McGraw-Hill, Inc., 1987.
- 25 M. Suzuki. Correlation inequalities and phase transition in the generalized X-Y model. *Journal of Mathematical Physics*, 14(7):837–838, 1973.
- 26 M. Talagrand. How much are increasing sets positively correlated? *Combinatorica*, 16(2):243–258, 1996.



# Shrinkage of Decision Lists and DNF Formulas

Benjamin Rossman

Duke University, Durham, NC, USA

benjamin.rossman@duke.edu

---

## Abstract

We establish nearly tight bounds on the expected shrinkage of decision lists and DNF formulas under the  $p$ -random restriction  $\mathbf{R}_p$  for all values of  $p \in [0, 1]$ . For a function  $f$  with domain  $\{0, 1\}^n$ , let  $\text{DL}(f)$  denote the minimum size of a decision list that computes  $f$ . We show that

$$\mathbb{E}[\text{DL}(f|_{\mathbf{R}_p})] \leq \text{DL}(f)^{\log_{2/(1-p)}(\frac{1+p}{1-p})}.$$

For example, this bound is  $\sqrt{\text{DL}(f)}$  when  $p = \sqrt{5} - 2 \approx 0.24$ . For Boolean functions  $f$ , we obtain the same shrinkage bound with respect to DNF formula size plus 1 (i.e., replacing  $\text{DL}(\cdot)$  with  $\text{DNF}(\cdot) + 1$  on both sides of the inequality).

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Circuit complexity

**Keywords and phrases** shrinkage, decision lists, DNF formulas

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.70

**Acknowledgements** I am grateful to the anonymous referees of ITCS 2021 for their valuable comments and to the authors of [17], Shachar Lovett, Kewen Wu and Jiapeng Zhang, for stimulating conversations related to this work.

## 1 Introduction

Random restrictions are a powerful tool in circuit complexity and the analysis of Boolean functions. A *restriction* is a partial assignment to the input bits of a function  $f$  on the hypercube  $\{0, 1\}^n$ . For a parameter  $p \in [0, 1]$ , the  $p$ -random restriction  $\mathbf{R}_p$  independently leaves each input bit free with probability  $p$  and otherwise assigns it to 0 or 1 with equal probability. We denote by  $f|_{\mathbf{R}_p}$  the function obtained from  $f$  by restricting its inputs to the subcube of  $\{0, 1\}^n$  that correspond to  $\mathbf{R}_p$ .

Random restrictions are known to reduce the complexity of functions in simple models of computations, such as decision trees (DT), decision lists (DL), DNF formulas (DNF), and DeMorgan formulas ( $\mathcal{L}$ ); the symbols in parentheses are notation for the corresponding size measures (see Section 2 for definitions). With respect to DeMorgan formula leaf-size  $\mathcal{L}$ , it is easy to see that  $\mathcal{L}(f|_{\mathbf{R}_p})$  has expectation at most  $p \cdot \mathcal{L}(f)$ . (This follows by linearity of expectation from the observation that each input literal in a minimal formula for  $f$  is eliminated by  $\mathbf{R}_p$  with probability  $p$ .) Subbotovskaya [25] was the first to show that the expected shrinkage factor is in fact significantly smaller than  $p$  (she showed an upper bound  $O(p^{3/2})$  for  $p \geq 1/\mathcal{L}(f)^{2/3}$ ). A subsequent line of results [1, 14, 19, 11, 26], culminating in an  $p^{2-o(1)}$  bound of Håstad [11] and a low-order improvement by Tal [26], eventually established an asymptotically tight bound:

► **Theorem 1** (Shrinkage of DeMorgan formulas [26]). *For all Boolean functions  $f$ ,*

$$\mathbb{E}[\mathcal{L}(f|_{\mathbf{R}_p})] = O(p^2 \mathcal{L}(f) + p\sqrt{\mathcal{L}(f)}).$$



© Benjamin Rossman;

licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 70; pp. 70:1–70:14

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

The constant 2 in the exponent  $p$  in Theorem 1 is known as the “shrinkage exponent” of DeMorgan formulas. Shrinkage under  $\mathbf{R}_p$  has also been studied for restricted types of formulas, namely read-once, monotone, and bounded-depth ( $\text{AC}^0$ ). It was shown in [5, 13] that read-once formulas have shrinkage exponent  $\log_{\sqrt{5}-1}(2) \approx 3.27$ . The shrinkage exponent of monotone formulas is between 2 and  $\log_{\sqrt{5}-1}(2)$  and conjectured to equal the latter; determining the exact constant is a longstanding question (Open Problem 4). In the  $\text{AC}^0$  setting (bounded-depth formulas with unbounded AND and OR gates), it is known that depth- $d$  formulas with fan-in  $m$  shrink to expected size  $O(1)$  under  $\mathbf{R}_p$  when  $p$  is  $O(1/\log m)^{d-1}$  [22]. However, it is open to determine the shrinkage rate for larger  $p$ , particularly in the “mild random restriction” regime where  $p$  is  $\Omega(1)$  or  $1 - o(1)$  (Open Question 2).

The results of this paper give nearly tight bounds on the shrinkage under  $\mathbf{R}_p$  of depth-2 formulas (also known as DNF and CNF formulas), as well as the more general computational model of *decision lists*. Before stating our main result, it is instructive to first consider shrinkage in the simpler model of *decision trees*. For a function  $f$  on the hypercube (with domain  $\{0, 1\}^n$  and arbitrary range), we denote by  $\text{DT}(f)$  the minimum number of leaves (i.e., output nodes) in a decision tree that computes  $f$ . The following bound is shown by straightforward induction on  $\text{DT}(f)$ . (I believe this bound is probably folklore, but could not find a reference so have included the short proof in Section 3.1.)

► **Theorem 2** (Shrinkage of decision trees). *For all functions  $f$  on the hypercube,*

$$\mathbb{E}[\text{DT}(f|_{\mathbf{R}_p})] \leq \text{DT}(f)^{\log_2(1+p)}.$$

*This bound holds with equality when  $f$  is a parity function.*

Decision lists are a natural computational model that has been studied in many contexts [3, 4, 16, 9, 21]. A *decision list of size  $m$*  is a sequence  $L = ((C_1, b_1), \dots, (C_m, b_m))$  where  $b_1, \dots, b_m$  are arbitrary output values and  $C_1, \dots, C_m$  are conjunctive clauses (ANDs of literals) such that  $C_1 \vee \dots \vee C_m$  is a tautology.<sup>1</sup>  $L$  computes a function on the hypercube as follows: on input  $x \in \{0, 1\}^n$ , the output is  $b_i$  for the first index  $i \in [m]$  such that  $C_i(x)$  is satisfied. We denote by  $\text{DL}(f)$  the minimum size of a decision list that computes  $f$ .

Decision lists are a generalization decision trees: every decision tree is equivalent to a decision list of the same size, and thus  $\text{DL}(f) \leq \text{DT}(f)$  for all functions  $f$  on the hypercube.<sup>2</sup> Boolean decision lists, in which  $b_1, \dots, b_m \in \{0, 1\}$ , are moreover a generalization of both DNF and CNF formulas. In particular, *DNF formulas* are the special case where  $b_1 = \dots = b_{m-1} = 1$  and  $b_m = 0$ . Following custom, we count the *size* of a DNF formula as  $m - 1$  instead of  $m$ , and thus  $\text{DL}(f) \leq \text{DNF}(f) + 1$  for all Boolean functions  $f$ .

Despite decision lists and DNF/CNF formulas being more complex computational models than decision trees, our main result shows that they shrink at a similar rate under  $\mathbf{R}_p$ .

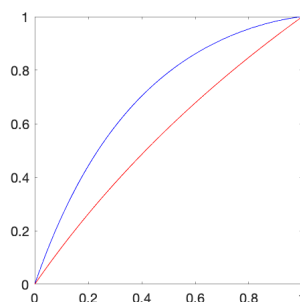
<sup>1</sup> In other words, every input  $x \in \{0, 1\}^n$  satisfies at least one of  $C_1, \dots, C_m$ . Without loss of generality,  $C_m$  may be chosen as the empty (always true) conjunctive clause  $\top$ . We allow  $C_1 \vee \dots \vee C_m$  to be an arbitrary tautology in order to more naturally define the class of *orthogonal* decision lists later on in Section 3.3.

<sup>2</sup> The name “decision list” elsewhere commonly refers to (what we call) width-1 decision trees, in which each clause is a single literal (i.e., an input variable  $x_i$  or its negation  $\bar{x}_i$ ). Whereas unbounded-width decision lists are a generalization decision trees, width-1 decision lists are instead a special case.

► **Theorem 3** (Shrinkage of decision lists and DNF formulas). *For all functions  $f$  on the hypercube,*

$$\mathbb{E}[\text{DL}(f|_{\mathbf{R}_p})] \leq \text{DL}(f)^{\gamma(p)} \quad \text{where} \quad \gamma(p) := \log_{\frac{2}{1-p}}\left(\frac{1+p}{1-p}\right).$$

*If  $f$  is Boolean, then also  $\mathbb{E}[\text{DNF}(f|_{\mathbf{R}_p}) + 1] \leq (\text{DNF}(f) + 1)^{\gamma(p)}$  (and similarly for  $\text{CNF}(\cdot) + 1$ ).*



■ **Figure 1** Plots of  $\gamma(p) := \log_{\frac{2}{1-p}}\left(\frac{1+p}{1-p}\right)$  (blue) and  $\log_2(1+p)$  (red).

Note that  $\gamma : [0, 1] \rightarrow [0, 1]$  is an increasing function with  $\gamma(0) = 0$  and  $\gamma(1) = 1$  (see Figure 1). The bound of Theorem 3 is thus nontrivial for all values of  $p \in (0, 1)$ . This bound is moreover close to optimal:  $\log_2(1+p)$  is a lower bound on the best possible function  $\gamma(p)$  (Section 3.4). As corollaries, we obtain additional bounds  $\text{ODL}(f)^{\gamma(p)}$  and  $\text{wODL}(f)^{\gamma(p)}$  on the shrinkage of orthogonal and weakly orthogonal decision lists (Corollary 14), as well as  $(\mathcal{L}_2(f) + 1)^{\gamma(2p)}$  for depth-2 formula leaf-size (Corollary 18).

Theorem 3 yields the following bounds for particular settings of  $p$  in terms of  $m = \text{DL}(f)$ :

$$\mathbb{E}[\text{DL}(f|_{\mathbf{R}_p})] \leq \begin{cases} 2 & \text{for } p = O\left(\frac{1}{\log m}\right), \\ \sqrt{m} & \text{for } p = \sqrt{5} - 2 \approx 0.24, \\ m/2 & \text{for } p = 1 - O\left(\frac{\log \log m}{\log m}\right), \\ m - 1 & \text{for } p = 1 - O\left(\frac{\log m}{m}\right). \end{cases}$$

For small  $p = O(1/\log m)$ , a variant of Håstad’s Switching Lemma (discussed below) actually implies a stronger inequality  $\mathbb{E}[\text{DT}(f|_{\mathbf{R}_p})] \leq 2$  with DT in place of DL (Corollary 6). Theorem 3 is mainly interesting for larger values of  $p$ . In particular, the “mild random restriction” regime when  $p$  is  $\Omega(1)$  or  $1 - o(1)$  has important applications in pseudorandomness [8, 20], DNF sparsification [7, 17] and hypercontractivity [18].

## 1.1 Switching lemmas and size measures vs. width/depth measures

We have so far discussed the shrinkage of various complexity measures under the  $p$ -random restriction  $\mathbf{R}_p$ . The switching lemmas stated below can be viewed as apples-to-oranges shrinkage results that bound one complexity measure on  $f|_{\mathbf{R}_p}$  in terms of another complexity measure on  $f$ . Here there is a useful distinction between “size measures” DT, DL, DNF and their corresponding “width/depth measures”, denoted by  $\text{DT}_{\text{depth}}$ ,  $\text{DL}_{\text{width}}$ ,  $\text{DNF}_{\text{width}}$ . Width/depth measures are typically related to the logarithm of size measures: functions

with size complexity  $m$  are approximable by (or in some cases equivalent to) functions with width/depth complexity  $O(\log m)$ . Håstad's Switching Lemma [10] gives a tail bound on the decision tree size of  $f|_{\mathbf{R}_p}$  in terms of the decision list width of  $f$ .<sup>3</sup>

► **Theorem 4** (Switching Lemma [10]). *For all functions  $f$  on the hypercube and  $t \in \mathbb{N}$ ,*

$$\mathbb{P}[\text{DT}_{\text{depth}}(f|_{\mathbf{R}_p}) \geq t] \leq O(p \cdot \text{DL}_{\text{width}}(f))^t.$$

A variant of the Switching Lemma with  $\log \text{DL}(f)$  in place of  $\text{DL}_{\text{width}}(f)$  was proved in [22].

► **Theorem 5** (Switching Lemma in terms of decision list size [22]). *For every function  $f$  on the hypercube and  $t \in \mathbb{N}$ ,*

$$\mathbb{P}[\text{DT}_{\text{depth}}(f|_{\mathbf{R}_p}) \geq t] \leq O(p \cdot \log \text{DL}(f))^t.$$

We remark that Theorem 5 follows directly from Theorem 4 for  $t \leq O(\log \text{DL}(f))$  (by the standard width reduction argument), but not for larger  $t$ . Obtaining a tail bound for all  $t \in \mathbb{N}$  is essentially to the following:

► **Corollary 6** (Decision tree size of decision lists). *For all functions  $f$  on  $\{0,1\}^n$ ,*

$$\mathbb{E}[\text{DT}(f|_{\mathbf{R}_p})] \leq 2 \quad \text{and} \quad \text{DT}(f) \leq O(2^{(1-p)^n}) \quad \text{where} \quad p = O(1/\log \text{DL}(f)).$$

As previously mentioned, Corollary 6 strengthens the bound  $\mathbb{E}[\text{DL}(f|_{\mathbf{R}_p})] \leq 2$  for  $p = O(1/\log \text{DL}(f))$  that follows from Theorem 3 (albeit for  $p$  that is a constant factor smaller). However, note that Corollary 6 is trivial for  $p$  above  $\Omega(1/\log \text{DL}(f))$ . A different switching lemma for large  $p$  (even  $1 - o(1)$ ) in terms of  $\text{DNF}_{\text{width}}(f)$  was introduced by Segerlind, Buss and Impagliazzo [24] and quantitatively improved by Razborov [20]. It is unclear if these switching lemmas for “mild random restriction” have analogues in terms of  $\log \text{DL}(f)$ ; if so, that might entail a shrinkage bound for  $\text{DL}$  that is nontrivial for all  $p \in (0, 1)$ , although potentially weaker than Theorem 3.

Our proof of Theorem 3 involves an application of Jensen's inequality with respect to a certain carefully defined probability distribution on the set of clauses in a decision list  $L$ . This distribution is related to (but not identical to) the distribution of the first satisfied clause of  $L$  under a uniform random input. A similar convexity argument appears in the proof of Theorem 5 in [22]. A second key idea, the notion of “useful indices” of  $L$  under a restriction  $\rho$ , comes from a recent paper of Lovett, Wu and Zhang [17] who proved the following result as the main lemma in establishing tight bound on the sparsification of bounded-width decision lists.

► **Theorem 7** (Decision list shrinkage in terms of width [17]). *For every function  $f$  on the hypercube,*

$$\mathbb{E}[\text{DL}(f|_{\mathbf{R}_p})] \leq \left( \frac{4}{1-p} \right)^{\text{DL}_{\text{width}}(f)}.$$

Note that our main result, Theorem 3, stands in relation to Theorem 7 just as Theorem 5 does to Theorem 4: in both cases we are essentially replacing  $\text{DL}_{\text{width}}(f)$  with  $\log \text{DL}(f)$ .

<sup>3</sup> In its application to  $\text{AC}^0$  circuit lower bounds, Theorem 4 is usually stated (more narrowly) in the form

$$\mathbb{P}[\text{CNF}_{\text{width}}(f|_{\mathbf{R}_p}) \geq t] \leq O(p \cdot \text{DNF}_{\text{width}}(f))^t$$

for Boolean functions  $f$ . The name “Switching Lemma” refers to the conversion of a DNF formula to a CNF formula. The more general bound stated in Theorem 4 is implicit in proofs of [10].

## 1.2 Other related work

There are different ways to quantify the effect of random restrictions on complexity measures. Instead of bounding expectation, one may show that shrinkage occurs with high probability. For DeMorgan formulas, high probability shrinkage results were shown in [23, 15]. Shrinkage results and switching lemmas have also been studied for random restrictions other than  $\mathbf{R}_p$  (see [2]). Very interesting recent work of Filmus, Meir and Tal [6] extends the technique of Håstad [11] to obtain  $p^{2-o(1)}$  factor shrinkage bounds for DeMorgan formulas under a family of pseudorandom projections that generalize  $\mathbf{R}_p$ .

## 2 Preliminaries

Throughout this paper,  $p$  is an arbitrary parameter in  $[0, 1]$ . All inequalities involving  $p$  hold for all values in  $[0, 1]$ . We often use the special case of Jensen's inequality  $\mathbb{E}[X^c] \leq \mathbb{E}[X]^c$  where  $X$  is a nonnegative random variable and  $c \in [0, 1]$  (in particular, when  $c$  is  $\log_2(1+p)$  or  $\gamma(p)$ ). We write  $\mathbb{N}$  for the natural numbers  $\{0, 1, 2, \dots\}$ , and for  $m \in \mathbb{N}$ , we write  $[m]$  for  $\{1, \dots, m\}$ .

### 2.1 Functions and restrictions on the hypercube

*Function on the hypercube* refers to any function with domain  $\{0, 1\}^n$  where  $n$  is a positive integer. A *Boolean function* is a function on the hypercube with codomain  $\{0, 1\}$ . (The parameter  $n$  plays no role in most results in this paper, so we suppress its mention whenever possible.)

A *restriction* is a partial assignment of Boolean variables  $x_1, \dots, x_n$  to values 0 and 1; this is formally defined as a function  $\varrho : \{1, \dots, n\} \rightarrow \{0, 1, *\}$  where  $\varrho(i) = *$  signifies that  $x_i$  is left free by  $\varrho$ . We denote by  $\text{Stars}(\varrho) \subseteq [n]$  the set of free variables under  $\varrho$ . For a function  $f$  on the hypercube  $\{0, 1\}^n$  and a restriction  $\varrho$ , we denote by  $f|_{\varrho}$  the restricted function on the subcube  $\{0, 1\}^{\text{Stars}(\varrho)}$  defined in the obvious way:  $(f|_{\varrho})(y) = f(x)$  where  $x \in \{0, 1\}^n$  is the input with  $x_i = y_i$  if  $i \in \text{Stars}(\varrho)$  and  $x_i = \varrho(i)$  otherwise.

For  $p \in [0, 1]$ , the  $p$ -*random restriction*  $\mathbf{R}_p$  is the random restriction that independently leaves each variable  $x_i$  free with probability  $p$  and otherwise sets  $x_i$  to 0 or 1 with equal probability. Thus, for any particular restriction  $\varrho$ , we have  $\mathbb{P}[\mathbf{R}_p = \varrho] = p^{|\text{Stars}(\varrho)|}((1-p)/2)^{n-|\text{Stars}(\varrho)|}$ .

### 2.2 Complexity measures DL, DT, DNF, CNF and their width/depth versions

► **Definition 8** (DNF formulas). We first define literals, conjunctive clauses, and DNF formulas over  $n$  variables.

- A *literal* is a Boolean variable  $x_i$  or negated Boolean variable  $\overline{x_i}$  where  $i \in \{1, \dots, n\}$ .
- A *conjunctive clause* (a.k.a. *term*) is an expression  $C$  of the form  $\ell_1 \wedge \dots \wedge \ell_w$  where  $\ell_1, \dots, \ell_w$  are literals on disjoint variables. The parameter  $w$  is the *width* of  $C$ ; this may be any nonnegative integer. The conjunctive clause of width zero is denoted by  $\top$ .
- A *DNF formula* is an expression  $F$  of the form  $C_1 \vee \dots \vee C_m$  where  $C_1, \dots, C_m$  are conjunctive clauses. The parameter  $m$  is the *size* of  $F$ ; this may be any nonnegative integer. The DNF formula of size 0 is denoted by  $\perp$ . The *width* of  $F$  is defined as the maximum width of any  $C_i$ .
- *CNF formulas* are defined dually (with the roles of  $\vee$  and  $\wedge$  exchanged).

## 70:6 Shrinkage of Decision Lists and DNF Formulas

Every literal, conjunctive clause, and DNF formula computes a Boolean function  $\{0, 1\}^n \rightarrow \{0, 1\}$  in the usual way.

- A DNF formula  $F$  is a *tautology* if it computes the identically 1 function. Note that any DNF formula that includes the empty conjunctive clause  $\top$  is a tautology.

► **Definition 9** (Decision lists).

- A *decision list* is an expression  $L$  of the form  $((C_1, b_1), \dots, (C_m, b_m))$  where  $b_1, \dots, b_m$  are arbitrary output values (not necessarily Boolean) and  $C_1, \dots, C_m$  are conjunctive clauses such that  $C_1 \vee \dots \vee C_m$  is a tautology. The parameter  $m$  is the *size* of  $L$ ; this may be any positive integer. The *width* of  $C$  is defined as the maximum width of any  $C_i$ .

A decision list  $L$  computes a function  $\{0, 1\}^n \rightarrow \{b_1, \dots, b_m\}$  as follows: on input  $x$ , the output is  $b_\ell$  where  $\ell \in [m]$  is the minimum index such that  $C_\ell(x) = 1$ . (Note that the final clause  $C_m$  may be replaced by  $\top$  without changing the function computed by  $L$ .)

► **Definition 10** (Decision trees).

- A *decision tree* is a rooted binary tree  $T$  in which each leaf is labeled by an output value (not necessarily Boolean) and each non-leaf node is labeled by a variable  $x_i$ , with the edges to its two children labeled “ $x_i = 0$ ” and “ $x_i = 1$ ”. The *size* of  $T$  is the number of leaves; this may be any positive integer. The *depth* of  $T$  is the maximum number of non-leaf nodes on any root-to-leaf branch; this may be any nonnegative integer.

► **Definition 11** (Associated complexity measures). For a function  $f$  with domain  $\{0, 1\}^n$  (and arbitrary codomain), let

$\text{DT}(f) :=$  minimum size of a decision tree that computes  $f$ ,

$\text{DL}(f) :=$  minimum size of a decision list that computes  $f$ ,

When  $f$  is Boolean, we additionally define

$\text{DNF}(f) :=$  minimum size of a DNF formula that computes  $f$ ,

$\text{CNF}(f) :=$  minimum size of a CNF formula that computes  $f$ .

For constant functions  $\underline{0}$  and  $\underline{1}$ , note that  $\text{DNF}(\underline{0}) = 0$  and  $\text{DNF}(\underline{1}) = 1$  according to our definition, since  $\underline{0}$  is computed by the empty DNF formula, while  $\underline{1}$  is computed by the DNF formula with a single empty clause. Also note that  $\text{CNF}(f) = \text{DNF}(\neg f)$ .

Each of the above size measures has a corresponding width/depth measure. These are denoted by

$$\text{DT}_{\text{depth}}(f), \quad \text{DL}_{\text{width}}(f), \quad \text{DNF}_{\text{width}}(f), \quad \text{CNF}_{\text{width}}(f).$$

► **Proposition 12** (see [3, 16]). *These size measures satisfy the following inequalities for all Boolean functions:*

$$1 \leq \text{DL} \leq \left\{ \begin{array}{c} \text{DNF} + 1 \\ \text{CNF} + 1 \end{array} \right\} \leq \text{DNF} + \text{CNF} \leq \text{DT}.$$

*The corresponding width/depth measures satisfy:*

$$0 \leq \text{DL}_{\text{width}} \leq \left\{ \begin{array}{c} \text{DNF}_{\text{width}} \\ \text{CNF}_{\text{width}} \\ \lceil \log_2(\text{DT}) \rceil \end{array} \right\} \leq \text{DT}_{\text{depth}} \leq \text{DNF}_{\text{width}} \cdot \text{CNF}_{\text{width}}.$$

*The above inequalities that involve decision trees and decision lists also apply to non-Boolean functions on the hypercube.*

We introduce additional computational models later on: (weakly) orthogonal decision lists in Section 3.3 and  $AC^0$  formulas in Section 4.

### 3 Shrinkage of decision trees and decision lists

We prove Theorems 2 and 3 in Sections 3.1 and 3. We then discuss extensions of our shrinkage bound to (weakly) orthogonal decision lists in Section 3.3 and tightness of the bounds Section 3.4.

#### 3.1 Shrinkage of decision trees

**Proof of Theorem 2.** Let  $T$  be a decision tree (with arbitrary output values). We must show that

$$\mathbb{E}[\text{size}(T \upharpoonright \mathbf{R}_p)] \leq \text{size}(T)^{\log_2(1+p)}.$$

We argue by induction of the size of  $T$ . The inequality is trivial in the base case that  $T$  has size 1.

Assume  $T$  has size  $m \geq 2$ . Then  $T$  has the form “If  $x_i = 0$  then  $T_0$  else  $T_1$ ” where  $T_0, T_1$  are decision trees of size  $m_0, m_1 \geq 1$  with  $m_0 + m_1 = m$ . Without loss of generality,  $T_0$  and  $T_1$  never query  $x_i$ . We have

$$\begin{aligned} \mathbb{E}[\text{size}(T \upharpoonright \mathbf{R}_p)] &= p \mathbb{E}[\text{size}(T \upharpoonright \mathbf{R}_p) \mid \mathbf{R}_p(x_i) = *] \\ &\quad + \frac{1-p}{2} \left( \mathbb{E}[\text{size}(T_0 \upharpoonright \mathbf{R}_p) \mid \mathbf{R}_p(x_i) = 0] + \mathbb{E}[\text{size}(T_1 \upharpoonright \mathbf{R}_p) \mid \mathbf{R}_p(x_i) = 1] \right) \\ &= \frac{1+p}{2} \left( \mathbb{E}[\text{size}(T_0 \upharpoonright \mathbf{R}_p)] + \mathbb{E}[\text{size}(T_1 \upharpoonright \mathbf{R}_p)] \right) \\ &\leq \frac{1+p}{2} \left( (m_0)^{\log_2(1+p)} + (m_1)^{\log_2(1+p)} \right) \quad (\text{induction hypothesis}) \\ &\leq (1+p) \left( \frac{m}{2} \right)^{\log_2(1+p)} \quad (\text{Jensen's inequality}) \\ &= m^{\log_2(1+p)}. \end{aligned}$$

As for tightness of the bound: If  $f$  is a parity function  $f(x_1, \dots, x_k) = x_1 \oplus \dots \oplus x_k$ , then we have  $\text{DT}(f) = 2^k$  and

$$\begin{aligned} \mathbb{E}[\text{DT}(f \upharpoonright \mathbf{R}_p)] &= \mathbb{E}[2^{\text{Bin}(k,p)}] = \sum_{i=0}^k 2^i \mathbb{P}[\text{Bin}(k,p) = i] \\ &= \sum_{i=0}^k \binom{k}{i} (2p)^i (1-p)^{k-i} = (1+p)^k = \text{DT}(f)^{\log_2(1+p)}. \blacktriangleleft \end{aligned}$$

#### 3.2 Shrinkage of decision lists

We now prove our main result on the shrinkage of decision lists and DNF formulas.

**Proof of Theorem 3.** Let  $f$  be any function on the hypercube and let  $p \in [0, 1]$ . (Note: Neither the hypercube dimension  $n$  nor the nature of output values of  $f$  play no role in our analysis.)



## 70:8 Shrinkage of Decision Lists and DNF Formulas

Let  $L = ((C_1, b_1), \dots, (C_m, b_m))$  be a decision list of minimum size that computes  $f$ , that is, with  $m = \text{DL}(f)$ . For  $\ell \in [m]$ , let  $|C_\ell|$  denote the width of the clause  $C_\ell$  (i.e., the number of literals in  $C_\ell$ ). Without loss of generality, we have  $|C_1|, \dots, |C_{m-1}| \geq 1$  and  $|C_m| = 0$  (i.e.,  $C_m$  is the empty clause  $\top$ ).

Following Lovett, Wu and Zhang [17], for a restriction  $\varrho$ , we define the set  $U(\varrho) \subseteq [m]$  of *useful indices of  $L$  under  $\varrho$*  by

$$U(\varrho) := \{\ell \in [m] : \exists x \text{ consistent with } \varrho \text{ s.t. } C_\ell(x) = 1 \text{ and } C_1(x) = \dots = C_{\ell-1}(x) = 0\}.$$

If  $U(\varrho) = \{\ell_1, \dots, \ell_t\}$  where  $1 \leq \ell_1 < \dots < \ell_t \leq m$ , then the restricted function  $f|_\varrho$  is computed by the decision list  $L|_\varrho$  defined by

$$L|_\varrho := ((C_{\ell_1}|_\varrho, b_{\ell_1}), \dots, (C_{\ell_t}|_\varrho, b_{\ell_t}))$$

where  $C_{\ell_i}|_\varrho$  is the sub-clause of  $C_{\ell_i}$  on the variables left unrestricted by  $\varrho$ . (Note that  $C_{\ell_1} \vee \dots \vee C_{\ell_t}$  is a tautology, so  $L|_\varrho$  is indeed a decision list.) Thus, we have

$$\text{DL}(f|_\varrho) \leq |U(\varrho)|. \tag{1}$$

For example, suppose  $m = 4$  and

$$C_1 = x_1 \wedge x_3, \quad C_2 = \overline{x_1} \wedge x_4, \quad C_3 = x_2 \wedge \overline{x_3}, \quad C_4 = \top.$$

For  $\varrho_1 := \{x_1 \mapsto 1\}$  (the restriction fixing  $x_1$  to 1 and leaving other variables free), we have

$$U(\varrho_1) = \{1, 3, 4\}, \quad L|_{\varrho_1} = ((x_3, b_1), (x_2 \wedge \overline{x_3}, b_3), (\top, b_4)).$$

For  $\varrho_2 := \{x_1 \mapsto 1, x_2 \mapsto 1\}$ , we have

$$U(\varrho_2) = \{1, 3\}, \quad L|_{\varrho_2} = ((x_3, b_1), (\overline{x_3}, b_3)).$$

In particular, the final clause  $C_4$  is not useful under  $\varrho_2$  (since any input consistent with  $\varrho_2$  satisfies  $C_1$  or  $C_3$ ).

Now comes a key definition: let  $\mu = (\mu_1, \dots, \mu_m)$  be the probability density vector (defining a probability distribution on  $[m]$ )

$$\mu_\ell := \mathbb{P}_{\varrho \sim \mathbf{R}_p} [\max(U(\varrho)) = \ell \text{ and } C_\ell|_\varrho \equiv 1] \quad \text{for } \ell \in [m-1],$$

$$\mu_m := \mathbb{P}_{\varrho \sim \mathbf{R}_p} [\max(U(\varrho)) = m \text{ or } C_{\max(U(\varrho))}|_\varrho \not\equiv 1].$$

Since events  $\max(U(\varrho)) = \ell$  are mutually exclusive, clearly we have  $\mu_1 + \dots + \mu_m = 1$ .

Note that  $\max(U(\varrho)) = \ell$  does not imply  $C_\ell|_\varrho \equiv 1$ , that is,  $\mu_\ell$  does not necessarily equal  $\mathbb{P}_{\varrho \sim \mathbf{R}_p} [\max(U(\varrho)) = \ell]$ . This is illustrated by the restriction  $\varrho_2$  in the above example, for which we have  $\max(U(\varrho_2)) = 3$ , yet  $C_3|_{\varrho_2} = \overline{x_3} \not\equiv 1$ . Restrictions  $\varrho_1$  and  $\varrho_2$  both contribute to probability mass  $\mu_4$ : in the case of  $\varrho_1$ , this is because  $\max(U(\varrho_1)) = 4$ , and in the case of  $\varrho_2$ , this is because  $C_{\max(U(\varrho_2))}|_{\varrho_2} \not\equiv 1$ .

For each  $\ell \in [m]$ , we have  $\mu_\ell \leq \mathbb{P}[C_\ell|_\varrho \equiv 1] = ((1-p)/2)^{|C_\ell|}$  and therefore

$$|C_\ell| \leq \log_{2/(1-p)}(1/\mu_\ell). \tag{2}$$

We require one more definition. For a restriction  $\varrho$  and a useful index  $\ell \in U(\varrho)$ , let  $\varrho^{(\ell)}$  be the restriction obtained by augmenting  $\varrho$  by the unique satisfying assignment for the clause  $C_\ell$ . That is,  $\varrho^{(\ell)}$  fixes a variable  $x_i$  to  $a \in \{0, 1\}$  if, and only if,  $\varrho$  fixes  $x_i$  to  $a$  or  $x_i = a$  in the satisfying assignment to  $C_\ell$ .

As in proofs of the Switching Lemma, we will use the fact that

$$\frac{\mathbb{P}[\mathbf{R}_p = \varrho]}{\mathbb{P}[\mathbf{R}_p = \varrho^{(\ell)}]} = \left( \frac{2p}{1-p} \right)^{|\text{Stars}(\varrho) \cap \text{Vars}(C_\ell)|} \quad (3)$$

since  $\varrho^{(\ell)}$  has exactly  $|\text{Stars}(\varrho) \cap \text{Vars}(C_\ell)|$  fewer unrestricted variables (“stars”) than  $\varrho$ .

As observed in [17], for every  $\ell \in U(\varrho)$ , we have  $U(\varrho^{(\ell)}) = U(\varrho) \cap [\ell]$  and therefore

$$\max(U(\varrho^{(\ell)})) = \ell \quad \text{and} \quad C_\ell \upharpoonright \varrho^{(\ell)} \equiv 1. \quad (4)$$

Thus,  $\varrho^{(\ell)}$  contributes to the probability mass  $\mu_\ell$ .

As a consequence of (3) and (4), we claim that for all  $\ell \in [m]$ ,

$$\mathbb{P}_{\varrho \sim \mathbf{R}_p} [\ell \in U(\varrho)] \leq \mu_\ell \left( \frac{1+p}{1-p} \right)^{|C_\ell|}. \quad (5)$$

In the case  $\ell = m$ , this follows from  $m \in U(\varrho) \Rightarrow \max(U(\varrho)) = m$ . For  $\ell \in [m-1]$ , this is shown as follows:

$$\begin{aligned} & \mathbb{P}_{\varrho \sim \mathbf{R}_p} [\ell \in U(\varrho)] \\ &= \sum_{S \subseteq \text{Vars}(C_\ell)} \mathbb{P}_{\varrho \sim \mathbf{R}_p} [\ell \in U(\varrho) \text{ and } \text{Stars}(\varrho) \cap \text{Vars}(C_\ell) = S] \\ &\stackrel{(4)}{\leq} \sum_{S \subseteq \text{Vars}(C_\ell)} \mathbb{P}_{\varrho \sim \mathbf{R}_p} [\ell = \max(U(\varrho^{(\ell)})) \text{ and } C_\ell \upharpoonright \varrho^{(\ell)} \equiv 1 \text{ and } \text{Stars}(\varrho) \cap \text{Vars}(C_\ell) = S] \\ &= \sum_{S \subseteq \text{Vars}(C_\ell)} \sum_{\varrho : \ell = \max(U(\varrho^{(\ell)})) \text{ and } C_\ell \upharpoonright \varrho^{(\ell)} \equiv 1 \text{ and } \text{Stars}(\varrho) \cap \text{Vars}(C_\ell) = S} \mathbb{P}[\mathbf{R}_p = \varrho] \\ &= \sum_{S \subseteq \text{Vars}(C_\ell)} \sum_{\sigma : \ell = \max(U(\sigma)) \text{ and } C_\ell \upharpoonright \sigma \equiv 1} \sum_{\varrho : \varrho^{(\ell)} = \sigma \text{ and } \text{Stars}(\varrho) \cap \text{Vars}(C_\ell) = S} \mathbb{P}[\mathbf{R}_p = \varrho] \\ &\stackrel{(3)}{=} \sum_{S \subseteq \text{Vars}(C_\ell)} \sum_{\sigma : \ell = \max(U(\sigma)) \text{ and } C_\ell \upharpoonright \sigma \equiv 1} \sum_{\varrho : \varrho^{(\ell)} = \sigma, \text{Stars}(\varrho) \cap \text{Vars}(C_\ell) = S} \left( \frac{2p}{1-p} \right)^{|S|} \mathbb{P}[\mathbf{R}_p = \sigma] \\ &= \sum_{S \subseteq \text{Vars}(C_\ell)} \left( \frac{2p}{1-p} \right)^{|S|} \sum_{\sigma : \ell = \max(U(\sigma)), C_\ell \upharpoonright \sigma \equiv 1} \mathbb{P}[\mathbf{R}_p = \sigma] \quad (\varrho \text{ is determined by } \sigma \text{ and } S) \\ &= \mu_\ell \sum_{S \subseteq \text{Vars}(C_\ell)} \left( \frac{2p}{1-p} \right)^{|S|} \quad (\text{definition of } \mu_\ell) \\ &= \mu_\ell \left( \frac{1+p}{1-p} \right)^{|C_\ell|} \quad (\text{binomial expansion of } (1 + \frac{2p}{1-p})^{|C_\ell|}). \end{aligned}$$

Finally, we obtain the shrinkage bound of Theorem 3 by the following calculation, which uses Jensen’s inequality in addition to the above observations:

## 70:10 Shrinkage of Decision Lists and DNF Formulas

$$\begin{aligned}
\mathbb{E}_{\varrho \sim \mathbf{R}_p} [\text{DL}(f|_{\varrho})] &\stackrel{(1)}{\leq} \mathbb{E}_{\varrho \sim \mathbf{R}_p} [|U(\varrho)|] = \sum_{\ell \in [m]} \mathbb{P}_{\varrho \sim \mathbf{R}_p} [\ell \in U(\varrho)] \\
&\stackrel{(5)}{=} \sum_{\ell \in [m]} \mu_{\ell} \left( \frac{1+p}{1-p} \right)^{|C_{\ell}|} \\
&\stackrel{(2)}{\leq} \sum_{\ell \in [m]} \mu_{\ell} \left( \frac{1+p}{1-p} \right)^{\log_2/(1-p)(1/\mu_{\ell})} \\
&= \mathbb{E}_{\ell \sim \mu} \left[ \left( \frac{1}{\mu_{\ell}} \right)^{\gamma(p)} \right] \quad (\text{def. of } \gamma(p) = \log_{\frac{2}{1-p}} \left( \frac{1+p}{1-p} \right)) \\
&\leq \left( \mathbb{E}_{\ell \sim \mu} \left[ \frac{1}{\mu_{\ell}} \right] \right)^{\gamma(p)} \quad (\text{Jensen's inequality}) \\
&= m^{\gamma(p)}.
\end{aligned}$$

Since  $m = \text{DL}(f)$ , this complete the proof of our bound on decision list shrinkage.

We shall now assume that  $f$  is Boolean and  $C_1 \vee \dots \vee C_m$  is a minimum size DNF formula computing  $f$ . Let  $L$  be the equivalent decision list  $((C_1, 1), \dots, (C_m, 1), (\top, 0))$  of size  $m + 1$ . The shrinkage bound

$$\mathbb{E}[\text{DNF}(f|_{\mathbf{R}_p}) + 1] \leq (\text{DNF}(f) + 1)^{\gamma(p)}$$

now follows from the above analysis, noting that  $\text{DNF}(f|_{\varrho}) + 1 \leq \text{size}(L|_{\varrho})$  for all restrictions  $\varrho$ .  $\blacktriangleleft$

### 3.3 Shrinkage of (weakly) orthogonal decision lists

► **Definition 13.** Let  $L = ((C_1, b_1), \dots, (C_m, b_m))$  be a decision list. We say that  $L$  is

- *orthogonal* if each input  $x$  satisfies exactly one of the conjunctive clauses  $C_1, \dots, C_m$ ,
- *weakly orthogonal* if each input  $x$  satisfies at most one of  $C_1, \dots, C_{m-1}$ .

(Note that if  $L$  is weakly orthogonal, then it remains so after replacing  $C_m$  with  $\top$ . In contrast, an orthogonal decision list has  $C_m = \top$  if and only if  $m = 1$ .)

For a function  $f$  on the hypercube, we denote by  $\text{wODL}(f)$  the minimum size of a (weakly) orthogonal decision list that computes  $f$ . These complexity measures lies in-between DL and DT:

$$\text{DL} \leq \text{wODL} \leq \text{ODL} \leq \text{DT}.$$

Our proof of Theorem 3 implies a shrinkage bound for ODL and wODL in the same way as for DNF + 1.

► **Corollary 14.** For every function  $f$  on the hypercube,

$$\mathbb{E}[\text{ODL}(f|_{\mathbf{R}_p})] \leq \text{ODL}(f)^{\gamma(p)} \quad \text{and} \quad \mathbb{E}[\text{wODL}(f|_{\mathbf{R}_p})] \leq \text{wODL}(f)^{\gamma(p)}.$$

This follows from the observation that if  $L$  is orthogonal, then so is  $L|_{\varrho}$  for any restriction  $\varrho$ , and if  $L$  is semi-orthogonal, then  $L|_{\varrho}$  is semi-orthogonal after replacing the final conjunctive clause with  $\top$ .

### 3.4 Lower bound on the optimal $\gamma(p)$

What is the optimal function  $\gamma(p)$  that may be chosen in the bound on decision list shrinkage of Theorem 3? We observe that  $\gamma(p)$  cannot be improved beyond  $\log_2(1+p)$ . The lower bound is given by a (non-Boolean) function  $f$  computed by a read-once decision tree of depth  $k$  and size  $2^k$ , in which each internal node queries a distinct variable and each leaf returns a distinct output value. For this  $f$ , we have  $\text{DL}(f) = 2^k$  and  $\mathbb{E}[\text{DL}(f \upharpoonright \mathbf{R}_p)] = (1+p)^k = \text{DL}(f)^{\log_2(1+p)}$ . The same function also shows that  $\gamma(p)$  in Corollary 14 cannot be improved beyond  $\log_2(1+p)$ . Since this function is not Boolean, it does not imply a lower bound on DNF shrinkage; however, a similar bound can be shown asymptotically by considering parity functions.

## 4 Shrinkage of $\text{AC}^0$ formulas

Our bound the shrinkage DNF and CNF formulas implies an (only slightly weaker) bound on the shrinkage of depth-2 formula leaf-size. We also discuss the relationship between leaf-size and a related size measure on  $\text{AC}^0$  formulas, the number of depth-1 gates.

► **Definition 15.** An  $\text{AC}^0$  formula is a formula composed unbounded fan-in AND and OR gates with inputs labeled by literals. We measure *depth* by the maximum number of gates on an input-to-output path; the expression “depth- $d$  formula” refers to an  $\text{AC}^0$  formula of depth at most  $d$ . As with DeMorgan formulas, the *leaf-size* of an  $\text{AC}^0$  formula is the number of leaves labeled by literals. An alternative size measure is the number of depth-1 gates (that have only literals as inputs). This number is at least half the total number of gates in any formula with no (useless) gates of fan-in 1.

For a Boolean function  $f$  and  $d \geq 2$ , we denote by  $\mathcal{L}_d(f)$  the minimum leaf-size of depth- $d$  formula that computes  $f$ , and we denote by  $\mathcal{F}_d(f)$  the minimum number of depth-1 gates in a depth- $d$  formula that computes  $f$ . Note that  $\mathcal{L}_d(f) = 1$  iff  $f$  is a literal, and  $\mathcal{F}_d(f) = 1$  iff  $f$  is a nonempty conjunctive or disjunctive clause, and  $\mathcal{L}_d(f) = \mathcal{F}_d(f) = 0$  iff  $f$  is constant (hence computed by a single AND or OR gate with fan-in zero, which as a formula has no inputs and no depth-1 gates).

Finally, we denote by  $\mathcal{F}(f)$  the minimum number of depth-1 gates in an (unbounded depth, unbounded fan-in) formula that computes  $f$ .

Note that  $\mathcal{F}_2 = \min\{\text{DNF}, \text{CNF}\}$ . Theorem 3 therefore implies:

► **Corollary 16.** For all Boolean functions  $f$ ,

$$\mathbb{E}[\mathcal{F}_2(f \upharpoonright \mathbf{R}_p) + 1] \leq (\mathcal{F}_2(f) + 1)^{\gamma(p)}.$$

Over  $n$ -variable Boolean functions, clearly  $\mathcal{F}_d \leq \mathcal{L}_d \leq n \cdot \mathcal{F}_d$  and  $\mathcal{F} \leq \mathcal{L} \leq n \cdot \mathcal{F}$ . The next lemma shows that, under a  $1/2$ -random restriction,  $\mathcal{F}_d$  shrinks below  $\mathcal{L}_d$  and  $\mathcal{F}$  shrinks below  $\mathcal{L}$  (independent of  $n$ ).

► **Lemma 17.** For all Boolean functions  $f$  and  $d \geq 2$ ,

$$\mathbb{E}[\mathcal{L}_d(f \upharpoonright \mathbf{R}_{1/2})] \leq \mathcal{F}_d(f) \quad \text{and} \quad \mathbb{E}[\mathcal{L}(f \upharpoonright \mathbf{R}_{1/2})] \leq \mathcal{F}(f).$$

**Proof.** Let  $F$  be a [depth- $d$ ]  $\text{AC}^0$  formula that computes  $f$  using the minimum number of depth-1 gates. By linearity of expectation, it suffices to show that each depth-1 subformula of  $F$  (i.e., conjunctive or disjunctive clause) has expected leaf-size at most 1 under  $\mathbf{R}_{1/2}$ . Indeed, for any  $k \geq 1$  and  $p \in [0, 1]$ ,

## 70:12 Shrinkage of Decision Lists and DNF Formulas

$$\mathbb{E}[\mathcal{L}(\text{AND}_k \upharpoonright \mathbf{R}_p)] = \mathbb{E}[\mathcal{L}(\text{OR}_k \upharpoonright \mathbf{R}_p)] = \sum_{j=0}^k j \binom{k}{j} p^j \left(\frac{1-p}{2}\right)^{k-j} = kp \left(\frac{1-p}{2}\right)^{k-1}.$$

When  $p = \frac{1}{2}$ , we have  $\frac{k}{2} \left(\frac{3}{4}\right)^{k-1} < 1$  for all  $k \geq 1$ . ◀

Using Lemma 17, we obtain the following bound on the shrinkage of depth-2 formula leaf-size  $\mathcal{L}_2$ , which has a slightly worse exponent  $\gamma(2p)$  compared to  $\gamma(p)$  for  $\mathcal{F}_2$  in Corollary 16.

► **Corollary 18** (Shrinkage of depth-2 formula leaf-size). *For all Boolean functions  $f$ ,*

$$\mathbb{E}[\mathcal{L}_2(f \upharpoonright \mathbf{R}_p) + 1] \leq (\mathcal{L}_2(f) + 1)^{\gamma(2p)}.$$

**Proof.** Viewing  $\mathbf{R}_p$  as a composition of  $\mathbf{R}_{1/2}$  (first) and  $\mathbf{R}_{2p}$  (second), we have

$$\begin{aligned} \mathbb{E}[\mathcal{L}_2(f \upharpoonright \mathbf{R}_p) + 1] &= \mathbb{E}_{\varrho \sim \mathbf{R}_{2p}} \left[ \mathbb{E}_{\sigma \sim \mathbf{R}_{1/2}} [\mathcal{L}_2((f \upharpoonright \varrho) \upharpoonright \sigma) + 1] \right] \\ &\leq \mathbb{E}_{\varrho \sim \mathbf{R}_{2p}} [\mathcal{F}_2(f \upharpoonright \varrho) + 1] && \text{(Lemma 17)} \\ &= (\mathcal{F}_2(f) + 1)^{\gamma(2p)} && \text{(Corollary 16)} \\ &\leq (\mathcal{L}_2(f) + 1)^{\gamma(2p)} && (\mathcal{F}_2 \leq \mathcal{L}_2). \end{aligned} \quad \blacktriangleleft$$

As an additional consequence of Lemma 17, we observe that  $\mathcal{F}$  has the same expected shrinkage factor (up to a constant factor) as DeMorgan leaf-size  $\mathcal{L}$ .

► **Corollary 19** (Shrinkage of unbounded fan-in, unbounded depth formulas). *For all Boolean functions  $f$ ,*

$$\mathbb{E}[\mathcal{F}(f \upharpoonright \mathbf{R}_p)] = O(p^2 \mathcal{F}(f) + p\sqrt{\mathcal{F}(f)}).$$

**Proof.** Assume  $p \leq 1/2$ , since the bound is trivial otherwise. Viewing  $\mathbf{R}_p$  as a composition of  $\mathbf{R}_{2p}$  (first) and  $\mathbf{R}_{1/2}$  (second), we have

$$\begin{aligned} \mathbb{E}[\mathcal{F}(f \upharpoonright \mathbf{R}_p)] &= \mathbb{E}_{\sigma \sim \mathbf{R}_{1/2}} \left[ \mathbb{E}_{\varrho \sim \mathbf{R}_{2p}} [\mathcal{F}((f \upharpoonright \sigma) \upharpoonright \varrho)] \right] \\ &\leq \mathbb{E}_{\sigma \sim \mathbf{R}_{1/2}} \left[ \mathbb{E}_{\varrho \sim \mathbf{R}_{2p}} [\mathcal{L}((f \upharpoonright \sigma) \upharpoonright \varrho)] \right] && (\mathcal{F} \leq \mathcal{L}) \\ &= \mathbb{E}_{\sigma \sim \mathbf{R}_{1/2}} \left[ O(4p^2 \mathcal{L}(f \upharpoonright \sigma) + 2p\sqrt{\mathcal{L}(f \upharpoonright \sigma)}) \right] && \text{(Theorem 1)} \\ &= O\left(p^2 \mathbb{E}_{\sigma \sim \mathbf{R}_{1/2}} [\mathcal{L}(f \upharpoonright \sigma)] + p\sqrt{\mathbb{E}_{\sigma \sim \mathbf{R}_{1/2}} [\mathcal{L}(f \upharpoonright \sigma)]}\right) && \text{(Jensen's inequality)} \\ &= O(p^2 \mathcal{F}(f) + p\sqrt{\mathcal{F}(f)}) && \text{(Lemma 17)}. \end{aligned} \quad \blacktriangleleft$$

## 5 Open problems

We conclude by mentioning some questions raised by this work.

► **Open Problem 1.** *Determine the optimal function  $\gamma_{\text{DL}}(p)$  in Theorem 3. We have shown that*

$$\log_2(1+p) = \gamma_{\text{DT}}(p) \leq \gamma_{\text{DL}}(p) \leq \log_{\frac{2}{1-p}}\left(\frac{1+p}{1-p}\right).$$

*A simpler problem is to determine the least constant  $C_{\text{DL}}$  such that  $\mathbb{E}[\text{DL}(f \upharpoonright \mathbf{R}_p)] \leq O(\text{DL}(f)^{C_{\text{DL}} \cdot p})$ . It follows from our bounds that  $\frac{1}{\ln 2} = C_{\text{DT}} \leq C_{\text{DL}} \leq \frac{2}{\ln 2}$ . The same questions may be asked with respect to complexity measures ODL, wODL and DNF.*

► **Open Problem 2.** *Determine the shrinkage rate of depth- $d$   $\text{AC}^0$  formulas for  $d \geq 3$ . We expect that*

$$\mathbb{E}[\mathcal{L}_d(f \upharpoonright \mathbf{R}_p)] \leq \mathcal{L}_d(f)^{O(p^{1/(d-1)})}. \quad (6)$$

*Ideally the constant in this big- $O$  should not depend on  $d$ .*

We remark that inequality (6) is known to hold for small  $p = O(1/\log \mathcal{L}_d(f))^{d-1}$ , when the bound is  $O(1)$ . This can be shown using the (Multi-)Switching Lemma of Håstad [12]. It is also a direct consequence of the following result of the author [22], which generalizes Corollary 6 (on the decision tree size of decision lists) to  $\text{AC}^0$  formulas of any depth.

► **Theorem 20** (Decision tree size of  $\text{AC}^0$  formulas [22]). *For all functions  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  computable by depth- $d$   $\text{AC}^0$  formulas with fan-in  $m$  (and leaf-size at most  $nm^{d-1}$ ),*

$$\mathbb{E}[\text{DT}(f \upharpoonright \mathbf{R}_p)] \leq 2 \quad \text{and} \quad \text{DT}(f) \leq O(2^{(1-p)^n}) \quad \text{where} \quad p = O(1/\log m)^{d-1}.$$

A related question:

► **Open Problem 3.** *Prove a stronger version of Theorem 20 for depth- $d$   $\text{AC}^0$  formulas with  $m = \mathcal{F}_d(f)^{1/(d-1)}$  (instead of fan-in, which is larger for unbalanced formulas). Such a result could be helpful in proving the shrinkage bound (6).*

Finally, we repeat the longstanding question concerning shrinkage of monotone formulas:

► **Open Problem 4.** *Determine the shrinkage exponent of monotone formulas. That is, find the maximum constant  $\Gamma_m$  such that*

$$\mathbb{E}[\mathcal{L}_m(f \upharpoonright \mathbf{R}_p)] \leq O(p^{\Gamma_m - o(1)} \mathcal{L}_m(f) + 1)$$

*for all monotone Boolean functions  $f$ , where  $\mathcal{L}_m$  is monotone formula leaf-size. It is known that  $2 = \Gamma_{\text{DeMorgan}} \leq \Gamma_m \leq \Gamma_{\text{read-once}} = \log_{\sqrt{5}-1}(2) \approx 3.27$ , and the second inequality is believed to be tight [5, 13].*

---

## References

- 1 Alexander E Andreev. On a method for obtaining more than quadratic effective lower bounds for the complexity of  $\pi$ -schemes. *Moscow Univ. Math. Bull.*, 42(1):63–66, 1987.
- 2 Paul Beame. A switching lemma primer. Technical report, Technical Report UW-CSE-95-07-01, Department of Computer Science and Engineering, University of Washington, 1994.

- 3 Avrim Blum. Rank- $r$  decision trees are a subclass of  $r$ -decision lists. *Information Processing Letters*, 42(4):183–185, 1992.
- 4 Nader H Bshouty. A subexponential exact learning algorithm for dnf using equivalence queries. *Information Processing Letters*, 59(1):37–39, 1996.
- 5 Moshe Dubiner and Uri Zwick. How do read-once formulae shrink? *parity*, 2(1):63, 1993.
- 6 Yuval Filmus, Or Meir, and Avishay Tal. Shrinkage under random projections and cubic formula lower bounds for  $AC^0$ . In *12th Innovations in Theoretical Computer Science Conference (ITCS)*, 2021.
- 7 Parikshit Gopalan, Raghu Meka, and Omer Reingold. DNF sparsification and a faster deterministic counting algorithm. *Computational Complexity*, 22(2):275–310, 2013.
- 8 Parikshit Gopalan, Raghu Meka, Omer Reingold, Luca Trevisan, and Salil Vadhan. Better pseudorandom generators from milder pseudorandom restrictions. In *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*, pages 120–129. IEEE, 2012.
- 9 Thomas Hancock, Tao Jiang, Ming Li, and John Tromp. Lower bounds on learning decision lists and trees. *Information and Computation*, 126(2):114–122, 1996.
- 10 Johan Håstad. Almost optimal lower bounds for small depth circuits. In *Proc. 18th ACM Symposium on Theory of Computing*, pages 6–20, 1986.
- 11 Johan Håstad. The shrinkage exponent of de Morgan formulas is 2. *SIAM Journal on Computing*, 27(1):48–64, 1998.
- 12 Johan Håstad. On the correlation of parity and small-depth circuits. *SIAM Journal on Computing*, 43(5):1699–1708, 2014.
- 13 Johan Håstad, Alexander Razborov, and Andrew Yao. On the shrinkage exponent for read-once formulae. *Theoretical Computer Science*, 141(1-2):269–282, 1995.
- 14 Russell Impagliazzo and Noam Nisan. The effect of random restrictions on formula size. *Random Structures & Algorithms*, 4(2):121–133, 1993.
- 15 Ilan Komargodski and Ran Raz. Average-case lower bounds for formula size. In *Proceedings of the 45th ACM Symposium on Theory of Computing*, pages 171–180, 2013.
- 16 Matthias Krause. On the computational power of boolean decision lists. *Computational Complexity*, 14(4):362–375, 2006.
- 17 Shachar Lovett, Kewen Wu, and Jiapeng Zhang. Decision list compression by mild random restrictions. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 247–254, 2020.
- 18 Ryan O’Donnell. *Analysis of boolean functions*. Cambridge University Press, 2014.
- 19 Michael S Paterson and Uri Zwick. Shrinkage of de Morgan formulae under restriction. *Random Structures & Algorithms*, 4(2):135–150, 1993.
- 20 Alexander A Razborov. Pseudorandom generators hard for  $k$ -DNF resolution and polynomial calculus resolution. *Annals of Mathematics*, pages 415–472, 2015.
- 21 Ronald L Rivest. Learning decision lists. *Machine learning*, 2(3):229–246, 1987.
- 22 Benjamin Rossman. Criticality of Regular Formulas. In *34th Computational Complexity Conference (CCC 2019)*, volume 137 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 1:1–28, 2019. doi:10.4230/LIPIcs.CCC.2019.1.
- 23 Rahul Santhanam. Fighting perebor: New and improved algorithms for formula and QBF satisfiability. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 183–192. IEEE, 2010.
- 24 Nathan Segerlind, Sam Buss, and Russell Impagliazzo. A switching lemma for small restrictions and lower bounds for  $k$ -DNF resolution. *SIAM Journal on Computing*, 33(5):1171–1200, 2004.
- 25 Bella A. Subbotovskaya. Realizations of linear functions by formulas using  $+$ ,  $\cdot$ ,  $-$ . *Doklady Akademii Nauk SSSR*, 136(3):553–555, 1961.
- 26 Avishay Tal. Shrinkage of De Morgan formulae by spectral techniques. In *55th Annual IEEE Symposium on Foundations of Computer Science*, pages 551–560, 2014.



# Block Rigidity: Strong Multiplayer Parallel Repetition Implies Super-Linear Lower Bounds for Turing Machines

**Kunal Mittal**

Department of Computer Science, Princeton University, NJ, USA  
kmittal@cs.princeton.edu

**Ran Raz**

Department of Computer Science, Princeton University, NJ, USA  
ranr@cs.princeton.edu

---

## Abstract

We prove that a sufficiently strong parallel repetition theorem for a special case of multiplayer (multiprover) games implies super-linear lower bounds for multi-tape Turing machines with advice. To the best of our knowledge, this is the first connection between parallel repetition and lower bounds for time complexity and the first major potential implication of a parallel repetition theorem with more than two players.

Along the way to proving this result, we define and initiate a study of *block rigidity*, a weakening of Valiant’s notion of *rigidity* [36]. While rigidity was originally defined for matrices, or, equivalently, for (multi-output) linear functions, we extend and study both rigidity and block rigidity for general (multi-output) functions. Using techniques of Paul, Pippenger, Szemerédi and Trotter [28], we show that a block-rigid function cannot be computed by multi-tape Turing machines that run in linear (or slightly super-linear) time, even in the non-uniform setting, where the machine gets an arbitrary advice tape.

We then describe a class of multiplayer games, such that, a sufficiently strong parallel repetition theorem for that class of games implies an explicit block-rigid function. The games in that class have the following property that may be of independent interest: for every random string for the verifier (which, in particular, determines the vector of queries to the players), there is a unique correct answer for each of the players, and the verifier accepts if and only if all answers are correct. We refer to such games as *independent games*. The theorem that we need is that parallel repetition reduces the value of games in this class from  $v$  to  $v^{\Omega(n)}$ , where  $n$  is the number of repetitions.

As another application of block rigidity, we show conditional size-depth tradeoffs for boolean circuits, where the gates compute arbitrary functions over large sets.

**2012 ACM Subject Classification** Theory of computation → Circuit complexity

**Keywords and phrases** Block-rigidity, Matrix Rigidity, Parallel Repetition, Size-depth tradeoffs, Turing Machines

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.71

**Funding** Research supported by the Simons Collaboration on Algorithms and Geometry, by a Simons Investigator Award and by the National Science Foundation grants No. CCF-1714779, CCF-2007462.

## 1 Introduction

We study relations between three seemingly unrelated topics: parallel repetition of multiplayer games, a variant of Valiant’s notion of rigidity, that we refer to as block rigidity, and proving super-linear lower bounds for Turing machines with advice.



© Kunal Mittal and Ran Raz;  
licensed under Creative Commons License CC-BY  
12th Innovations in Theoretical Computer Science Conference (ITCS 2021).  
Editor: James R. Lee; Article No. 71; pp. 71:1–71:15



Leibniz International Proceedings in Informatics  
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

## 1.1 Super-Linear Lower Bounds for Turing Machines

Deterministic multi-tape Turing machines are the standard model of computation for defining time-complexity classes. Lower bounds for the running time of such machines are known by the time-hierarchy theorem [18], using a diagonalization argument. Moreover, the seminal work of Paul, Pippenger, Szemerédi and Trotter gives a separation of non-deterministic linear time from deterministic linear time, for multi-tape Turing machines [28] (using ideas from [20] and [27]). That is, it shows that  $\text{DTIME}(n) \subsetneq \text{NTIME}(n)$ . This result has been slightly improved to give  $\text{DTIME}(n\sqrt{\log^* n}) \subsetneq \text{NTIME}(n\sqrt{\log^* n})$  [33].

However, the above mentioned lower bounds do not hold in the non-uniform setting, where the machines are allowed to use arbitrary advice depending on the length of the input. In the non-uniform setting, no super-linear lower bound is known for the running time of deterministic multi-tape Turing machines. Moreover, such bounds are not known even for a multi-output function.

## 1.2 Block Rigidity

The concept of matrix rigidity was introduced by Valiant as a means to prove super-linear lower bounds against circuits of logarithmic depth [36]. Since then, it has also found applications in communication complexity [32] (see also [39, 25]). We extend Valiant's notion of rigid matrices to the concept of *rigid functions*, and further to *block-rigid functions*. We believe that these notions are of independent interest. We note that block-rigidity is a weaker condition than rigidity and hence it may be easier to find explicit block-rigid functions. Further, our result gives a new application of rigidity.

Over a field  $\mathbb{F}$ , a matrix  $A \in \mathbb{F}^{n \times n}$  is said to be an  $(r, s)$ -rigid matrix if it is not possible to reduce the rank of  $A$  to at most  $r$ , by changing at most  $s$  entries in each row of  $A$ . Valiant showed that if  $A \in \mathbb{F}^{n \times n}$  is  $(\epsilon n, n^\epsilon)$ -rigid for some constant  $\epsilon > 0$ , then  $A$  is not computable by a linear-circuit of logarithmic depth and linear size. As in many problems in complexity, the challenge is to find explicit rigid matrices. By explicit, we mean that a polynomial time deterministic Turing machine should be able to output a rigid matrix  $A \in \mathbb{F}^{n \times n}$  on input  $1^n$ . The best known bounds on explicit rigid matrices are far from what is needed to get super-linear circuit lower bounds (see [15, 35, 34, 24, 26, 23, 2, 17, 1, 4]).

We extend the above definition to functions  $f : \{0, 1\}^n \rightarrow \{0, 1\}^n$ , by saying that  $f$  is *not* an  $(r, s)$ -rigid function if there exists a subset  $X \subseteq \{0, 1\}^n$  of size at least  $2^{n-r}$ , such that over  $X$ , each output bit of  $f$  can be written as a function of some  $s$  input bits (see Definition 7). By a simple counting argument (see Proposition 8), it follows that random functions are rigid with good probability.

We further extend this definition to what we call block-rigid functions (see Definition 11). For this, we'll consider vectors  $x \in \{0, 1\}^{nk}$ , which are thought of as composed of  $k$  blocks, each of size  $n$ . We say that a function  $f : \{0, 1\}^{nk} \rightarrow \{0, 1\}^{nk}$  is *not* an  $(r, s)$ -block-rigid function, if there exists a subset  $X \subseteq \{0, 1\}^{nk}$  of size at least  $2^{nk-r}$ , such that over  $X$ , each *output block* of  $f$  can be written as a function of some  $s$  *input blocks*.

We conjecture that it is possible to obtain large block-rigid functions, using smaller rigid functions. For a function  $f : \{0, 1\}^k \rightarrow \{0, 1\}^k$ , we define the function  $f^{\otimes n} : \{0, 1\}^{nk} \rightarrow \{0, 1\}^{nk}$  as follows. For each  $x = (x_{ij})_{i \in [n], j \in [k]} \in \{0, 1\}^{nk}$ , we define  $f^{\otimes n}(x)$  to be the vector obtained by applying  $f$  to  $(x_{i1}, \dots, x_{ik})$ , in place for each  $i \in [n]$  (see Definition 13).

► **Conjecture 1.** *There exists a universal constant  $c > 0$  such that the following is true. Let  $f : \{0, 1\}^k \rightarrow \{0, 1\}^k$  be an  $(r, s)$ -rigid function, and  $n \in \mathbb{N}$ . Then,  $f^{\otimes n} : \{0, 1\}^{nk} \rightarrow \{0, 1\}^{nk}$  is a  $(cnr, cs)$ -block-rigid function.*

We prove the following theorem. It is restated and proved as Theorem 28 in Section 5.

- **Theorem 2.** *Let  $t : \mathbb{N} \rightarrow \mathbb{N}$  be any function such that  $t(n) = \omega(n)$ . Assuming Conjecture 1, there exists an (explicitly given) function  $f : \{0, 1\}^* \rightarrow \{0, 1\}^*$  such that*
1. *On inputs  $x$  of length  $n$  bits, the output  $f(x)$  is of length at most  $n$  bits.*
  2. *The function  $f$  is computable by a multi-tape deterministic Turing machine that runs in time  $O(t(n))$  on inputs of length  $n$ .*
  3. *The function  $f$  is not computable by any multi-tape deterministic Turing machine that takes advice and runs in time  $O(n)$  on inputs of length  $n$ .*

More generally, we show that families of block-rigid functions cannot be computed by non-uniform Turing machines running in linear-time. This makes it interesting to find such families that are computable in polynomial time. The following theorem is restated as Theorem 29.

- **Theorem 3.** *Let  $k : \mathbb{N} \rightarrow \mathbb{N}$  be a function such that  $k(n) = \omega(1)$  and  $k(n) = 2^{o(n)}$ , and  $f_n : \{0, 1\}^{nk(n)} \rightarrow \{0, 1\}^{nk(n)}$  be a family of  $(\epsilon nk(n), \epsilon k(n))$ -block-rigid functions, for some constant  $\epsilon > 0$ . Let  $M$  be any multi-tape deterministic linear-time Turing machine that takes advice. Then, there exists  $n \in \mathbb{N}$ , and  $x \in \{0, 1\}^{nk(n)}$ , such that  $M(x) \neq f_n(x)$ .*

As another application, based on Conjecture 1, we show size-depth tradeoffs for boolean circuits, where the gates compute arbitrary functions over large (with respect to the input size) sets (see Section 6).

### 1.3 Parallel Repetition

In a  $k$ -player game  $\mathcal{G}$ , questions  $(x_1, \dots, x_k)$  are chosen from some joint distribution  $\mu$ . For each  $j \in [k]$ , player  $j$  is given  $x_j$  and gives an answer  $a_j$  that depends only on  $x_j$ . The players are said to win if their answers satisfy a fixed predicate  $V(x_1, \dots, x_k, a_1, \dots, a_k)$ . We note that  $V$  might be randomized, that is, it might depend on some random string that is sampled independently of  $(x_1, \dots, x_k)$ . The value of the game  $\text{val}(\mathcal{G})$  is defined to be the maximum winning probability over the possible strategies of the players.

It is natural to consider the parallel repetition  $\mathcal{G}^{\otimes n}$  of such a game  $\mathcal{G}$ . Now, the questions  $(x_1^{(i)}, \dots, x_k^{(i)})$  are chosen from  $\mu$ , independently for each  $i \in [n]$ . For each  $j \in [k]$ , player  $j$  is given  $(x_j^{(1)}, \dots, x_j^{(n)})$  and gives answers  $(a_j^{(1)}, \dots, a_j^{(n)})$ . The players are said to win if the answers satisfy  $V(x_1^{(i)}, \dots, x_k^{(i)}, a_1^{(i)}, \dots, a_k^{(i)})$  for every  $i \in [n]$ . The value of the game  $\text{val}(\mathcal{G}^{\otimes n})$  is defined to be the maximum winning probability over the possible strategies of the players. Note that the players are allowed to correlate their answers to different repetitions of the game.

Parallel repetition of games was first studied in [13], owing to its relation with multiprover interactive proofs [3]. It was hoped that the value  $\text{val}(\mathcal{G}^{\otimes n})$  of the repeated game goes down as  $\text{val}(\mathcal{G})^n$ . However, this is not the case, as shown in [14, 10, 11, 31].

A lot is known about parallel repetition of 2-player games. The, so called, parallel repetition theorem, first proved by Raz [30] and further simplified and improved by Holenstein [19], shows that if  $\text{val}(\mathcal{G}) < 1$ , then  $\text{val}(\mathcal{G}^{\otimes n}) \leq 2^{-\Omega(n/s)}$ , where  $s$  is the length of the answers given by the players. The bounds in this theorem were later made tight even for the case when the initial game has small value (see [8] and [5]).

Much less is known for  $k$ -player games with  $k \geq 3$ . Verbitsky [38] showed that if  $\text{val}(\mathcal{G}) < 1$ , then the value of the repeated game goes down to zero as  $n$  grows larger. The result shows a very weak rate of decay, approximately equal to  $\frac{1}{\alpha(n)}$ , where  $\alpha$  is the

inverse-Ackermann function, owing to the use of the density Hales-Jewett theorem (see [16] and [29]). A recent result by Dinur, Harsha, Venkat and Yuen [7] shows exponential decay, but only in the special case of what they call *expanding games*. This approach fails when the input to the players have strong correlations.

In this paper (see Section 4), we show that a sufficiently strong parallel repetition theorem for multiplayer games implies Conjecture 1. The following theorem is proved formally as Theorem 19 in Section 4.

► **Theorem 4.** *There exists a family  $\{\mathcal{G}_{\mathcal{S},k}\}$  of  $k$ -player games (where  $\mathcal{S}$  is some parameter), such that a strong parallel repetition theorem for all games in  $\{\mathcal{G}_{\mathcal{S},k}\}$  implies Conjecture 1.*

Although the games in this family do not fit into the framework of [7], they satisfy some very special properties. Every  $k$ -player game in the family satisfies the following:

1. The questions to the  $k$ -players are chosen as follows: First,  $k$  bits,  $x_1, \dots, x_k \in \{0, 1\}$ , are drawn uniformly and independently. Each of the  $k$ -players sees some subset of these  $k$ -bits.
2. The predicate  $V$  satisfies the condition that on fixing the bits  $x_1, \dots, x_k$ , there is a unique accepting answer for each player (independently of all other answers) and the verifier accepts if every player answers with the accepting answer. We refer to games that satisfy this property as *independent games*.

We believe that these properties may allow us to prove strong upper bounds on the value of parallel repetition of such games, despite our lack of understanding of multiplayer parallel repetition. The bounds that we need are that parallel repetition reduces the value of such games from  $v$  to  $v^{\Omega(n)}$ , where  $n$  is the number of repetitions (as is proved in [8] and [5] for 2-player games).

## 1.4 Open Problems

1. The main open problem is to make progress towards proving Conjecture 1, possibly using the framework of parallel repetition. The remarks after Theorem 28 mention some weaker statements that suffice for our applications. The examples of matrix-transpose and matrix-product in Section 8 also serve as interesting problems.
2. Our techniques, which are based on [28], heavily exploit the fact that the Turing machines have one-dimensional tapes. Time-space lower bounds for satisfiability in the case of multi-tape Turing machines with random access [12], and Turing machines with one  $d$ -dimensional tape [37], are known. Extending such results to the non-uniform setting is an interesting open problem.
3. The question of whether a rigid-matrix  $A \in \mathbb{F}_2^{n \times n}$  is rigid when seen as a function  $A : \{0, 1\}^n \rightarrow \{0, 1\}^n$  is very interesting (see Section 7). This question is closely related to a Conjecture of Jukna and Schnitger [21] on the linearization of depth-2 circuits. This is also related to the question of whether data structures for linear problems can be optimally linearized (see [9]). We note that there are known examples of linear problems for which the best known data-structures are non-linear, without any known linear data-structure achieving the same bounds (see [22]).

## 2 Preliminaries

Let  $\mathbb{N} = \{1, 2, 3, \dots\}$  be the set of all natural numbers. For any  $k \in \mathbb{N}$ , we use  $[k]$  to denote the set  $\{1, 2, \dots, k\}$ .

We use  $\mathbb{F}$  to denote an arbitrary finite field, and  $\mathbb{F}_2$  to denote the finite field on two elements.

Let  $x \in \{0, 1\}^k$ . For  $i \in [k]$ , we use  $x_i$  to denote the  $i^{\text{th}}$  coordinate of  $x$ . For  $S \subseteq [k]$ , we denote by  $x|_S$  the vector  $(x_i)_{i \in S}$ , which is the restriction of  $x$  to coordinates in  $S$ .

We also consider vectors  $x \in \{0, 1\}^{nk}$ , for some  $n \in \mathbb{N}$ . We think of these as composed of  $k$  blocks, each consisting of a vector in  $\{0, 1\}^n$ . That is,  $x = (x_{ij})_{i \in [n], j \in [k]}$ . By abuse of notation, for  $S \subseteq [k]$ , we denote by  $x|_S$  the vector  $(x_{ij})_{i \in [n], j \in S}$ , which is the restriction of  $x$  to the blocks indexed by  $S$ .

Let  $A \in \mathbb{F}^{nk \times nk}$  be an  $nk \times nk$  matrix. We think of  $A$  as a block-matrix consisting of  $k^2$  blocks, each block being an  $n \times n$  matrix. That is,  $A = (A_{ij})_{i, j \in [k]}$ , where for all  $i, j \in [k]$ ,  $A_{ij} \in \mathbb{F}^{n \times n}$ . For each  $i \in [k]$ , we call  $(A_{ij})_{j \in [k]}$  the  $i^{\text{th}}$  block-row of  $A$ .

For every  $n \in \mathbb{N}$ , we define  $\log^* n = \min\{\ell \in \mathbb{N} \cup \{0\} : \underbrace{\log_2 \log_2 \dots \log_2}_{\ell \text{ times}} n \leq 1\}$ .

### 3 Rigidity and Block Rigidity

#### 3.1 Rigidity

The concept of matrix rigidity was introduced by Valiant [36]. It is defined as follows.

► **Definition 5.** A matrix  $A \in \mathbb{F}^{n \times n}$  is said to be an  $(r, s)$ -rigid matrix if it cannot be written as  $A = B + C$ , where  $B$  has rank at most  $r$ , and  $C$  has at most  $s$  non-zero entries in each row.

Valiant [36] showed the existence of rigid matrices by a simple counting argument. For the sake of completeness, we include this proof.

► **Proposition 6.** For any constant  $0 < \epsilon \leq \frac{1}{8}$ , and any  $n \in \mathbb{N}$ , there exists a matrix  $A \in \mathbb{F}^{n \times n}$  that is an  $(\epsilon n, \epsilon n)$ -rigid matrix.

**Proof.** Fix any  $0 < \epsilon \leq \frac{1}{8}$ . We bound the number of  $n \times n$  matrices that are not  $(\epsilon n, \epsilon n)$ -rigid matrices.

1. Any  $n \times n$  matrix with rank at most  $r$  can be written as the product of an  $n \times r$  and an  $r \times n$  matrix. Hence, the number of matrices of rank at most  $\epsilon n$  is at most  $|\mathbb{F}|^{2\epsilon n^2} \leq |\mathbb{F}|^{\frac{n^2}{4}}$ .
2. The number of matrices that have at most  $\epsilon n$  non-zero entries in each row is at most

$$\left( \binom{n}{\epsilon n} |\mathbb{F}|^{\epsilon n} \right)^n \leq \left( \frac{e}{\epsilon} \right)^{\epsilon n^2} |\mathbb{F}|^{\epsilon n^2} = |\mathbb{F}|^{\epsilon n^2 (1 + \log_{|\mathbb{F}|} \frac{e}{\epsilon})} < |\mathbb{F}|^{\frac{3n^2}{4}}.$$

We used the binomial estimate  $\binom{n}{r} \leq \left(\frac{en}{r}\right)^r$ .

Since each matrix that is not an  $(r, s)$ -rigid matrix can be written as the sum of a matrix with rank at most  $r$ , and a matrix with at most  $s$  non-zero entries in each row, the number of matrices that are not  $(\epsilon n, \epsilon n)$ -rigid matrices is strictly less than  $|\mathbb{F}|^{\frac{n^2}{4}} \cdot |\mathbb{F}|^{\frac{3n^2}{4}} = |\mathbb{F}|^{n^2}$ , which is the total number of  $n \times n$  matrices. ◀

Observe that a matrix  $A \in \mathbb{F}_2^{n \times n}$  is not an  $(r, s)$ -rigid matrix if and only if there is a subspace  $X \subseteq \mathbb{F}_2^n$  of dimension at least  $n - r$ , and a matrix  $C$  with at most  $s$  non-zero entries in each row, such that  $Ax = Cx$  for all  $x \in X$ .

We use this formulation to extend the concept of rigidity to general functions.

► **Definition 7.** A function  $f : \{0, 1\}^n \rightarrow \{0, 1\}^n$  is said to be an  $(r, s)$ -rigid function if for every subset  $X \subseteq \{0, 1\}^n$  of size at least  $2^{n-r}$ , and subsets  $S_1, \dots, S_n \subseteq [n]$  of size  $s$ , and functions  $g_1, \dots, g_n : \{0, 1\}^s \rightarrow \{0, 1\}$ , there exists  $x \in X$  such that  $f(x) \neq (g_1(x|_{S_1}), g_2(x|_{S_2}), \dots, g_n(x|_{S_n}))$ .

Using a similar counting argument as in Proposition 6, we show the existence of rigid functions.

► **Proposition 8.** *For any constant  $0 < \epsilon \leq \frac{1}{8}$ , and any (large enough) integer  $n$ , there exists a function  $f : \{0, 1\}^n \rightarrow \{0, 1\}^n$  that is an  $(\epsilon n, \epsilon n)$ -rigid function.*

**Proof.** Fix any constant  $0 < \epsilon \leq \frac{1}{8}$ , and any integer  $n \geq \frac{2}{\epsilon}$ . We count the number of functions  $f : \{0, 1\}^n \rightarrow \{0, 1\}^n$  that are not  $(\epsilon n, \epsilon n)$ -rigid functions.

1. Note that in Definition 7, it is without loss of generality to assume that  $|X| = 2^{n-\epsilon n}$ . The number subsets  $X \subseteq \{0, 1\}^n$  of size  $2^{n-\epsilon n}$  is  $\binom{2^n}{2^{n-\epsilon n}} \leq (e2^{\epsilon n})^{2^{n-\epsilon n}} < 2^{2\epsilon n 2^{n-\epsilon n}} \leq 2^{\frac{n}{4} 2^{n-\epsilon n}}$ .
  2. The number of subsets  $S_1, \dots, S_n \subseteq [n]$  of size  $\epsilon n$  is at most  $\binom{n}{\epsilon n}^n < n^{\epsilon n^2} < 2^{\frac{n}{4} 2^{n-\epsilon n}}$ .
  3. The number of functions  $g_1, \dots, g_n : \{0, 1\}^{\epsilon n} \rightarrow \{0, 1\}$  is at most  $2^{n 2^{\epsilon n}} < 2^{\frac{n}{4} 2^{n-\epsilon n}}$ .
  4. The number of choices for values of  $f$  on  $\{0, 1\}^n \setminus X$  is at most  $2^{n(2^n - |X|)} \leq 2^{n(2^n - 2^{n-\epsilon n})}$ .
- Hence, the total number of functions  $f : \{0, 1\}^n \rightarrow \{0, 1\}^n$  that are not  $(\epsilon n, \epsilon n)$ -rigid functions is strictly less than  $\left(2^{\frac{n}{4} 2^{n-\epsilon n}}\right)^3 2^{n 2^n - n 2^{n-\epsilon n}} < 2^{n 2^n}$ . ◀

### 3.2 Block Rigidity

In this section, we introduce the notion of block rigidity.

► **Definition 9.** *A matrix  $A \in \mathbb{F}^{nk \times nk}$  is said to be an  $(r, s)$ -block-rigid matrix if it cannot be written as  $A = B + C$ , where  $B$  has rank at most  $r$ , and  $C$  has at most  $s$  non-zero matrices in each block-row.*

Observe that if  $A \in \mathbb{F}^{nk \times nk}$  is an  $(r, ns)$ -rigid matrix, then it is also  $(r, s)$ -block-rigid matrix. Combining this with Proposition 6, we get the following.

► **Observation 10.** *For any constant  $0 < \epsilon \leq \frac{1}{8}$ , and positive integers  $n, k$ , there exists an  $(\epsilon nk, \epsilon k)$ -block-rigid matrix  $A \in \mathbb{F}^{nk \times nk}$ .*

Following the definition of rigid-functions in Section 3.1, we define block-rigid functions as follows.

► **Definition 11.** *A function  $f : \{0, 1\}^{nk} \rightarrow \{0, 1\}^{nk}$  is said to be an  $(r, s)$ -block-rigid function if for every subset  $X \subseteq \{0, 1\}^{nk}$  of size at least  $2^{nk-r}$ , and subsets  $S_1, \dots, S_k \subseteq [k]$  of size  $s$ , and functions  $g_1, \dots, g_k : \{0, 1\}^{ns} \rightarrow \{0, 1\}^n$ , there exists  $x \in X$  such that  $f(x) \neq (g_1(x|_{S_1}), g_2(x|_{S_2}), \dots, g_k(x|_{S_k}))$ .*

Observe that if  $f : \{0, 1\}^{nk} \rightarrow \{0, 1\}^{nk}$  is an  $(r, ns)$ -rigid function, then it is also  $(r, s)$ -block-rigid function. Combining this with Proposition 8, we get the following.

► **Observation 12.** *For any constant  $0 < \epsilon \leq \frac{1}{8}$ , and (large enough) integers  $n, k$ , there exists an  $(\epsilon nk, \epsilon k)$ -block-rigid function  $f : \{0, 1\}^{nk} \rightarrow \{0, 1\}^{nk}$ .*

Note that  $n = 1$  in the definition of block-rigid matrices (functions) gives the usual definition of rigid matrices (functions). For our applications, we will mostly be interested in the case when  $n$  is much larger than  $k$ .

### 3.3 Rigidity Amplification

A natural question is whether there is a way to amplify rigidity. That is, given a rigid matrix (function), is there a way to obtain a larger matrix (function) which is rigid, or even block-rigid.

► **Definition 13.** Let  $f : \{0, 1\}^k \rightarrow \{0, 1\}^k$  be any function. Define  $f^{\otimes n} : \{0, 1\}^{nk} \rightarrow \{0, 1\}^{nk}$  as following. Let  $x = (x_{ij})_{i \in [n], j \in [k]} \in \{0, 1\}^{nk}$  and  $i \in [n], j \in [k]$ . The  $(i, j)^{\text{th}}$  coordinate of  $f^{\otimes n}(x)$  is defined to be the  $j^{\text{th}}$  coordinate of  $f(x_{i1}, x_{i2}, \dots, x_{ik})$ .

Basically, applying  $f^{\otimes n}$  on  $x \in \{0, 1\}^{nk}$  is the same as applying  $f$  on  $(x_{i1}, x_{i2}, \dots, x_{ik})$ , in place for each  $i \in [n]$ . For a linear function given by matrix  $A \in \mathbb{F}_2^{k \times k}$ , this operation corresponds to  $A \otimes I_n$ , where  $I_n$  is the  $n \times n$  identity matrix, and  $\otimes$  denotes the Kronecker product of matrices.

It is easy to see that if  $f$  is not rigid, then  $f^{\otimes n}$  is not block-rigid.

► **Observation 14.** Suppose  $f : \{0, 1\}^k \rightarrow \{0, 1\}^k$  is not an  $(r, s)$ -rigid function. Then  $f^{\otimes n}$  is not an  $(nr, s)$ -block-rigid function.

The converse of Observation 14 is more interesting. We believe that it is true, and restate Conjecture 1 below.

► **Conjecture 15.** There exists a universal constant  $c > 0$  such that the following is true. Let  $f : \{0, 1\}^k \rightarrow \{0, 1\}^k$  be an  $(r, s)$ -rigid function, and  $n \in \mathbb{N}$ . Then,  $f^{\otimes n} : \{0, 1\}^{nk} \rightarrow \{0, 1\}^{nk}$  is a  $(cnr, cs)$ -block-rigid function.

## 4 Parallel Repetition

In this section, we show an approach to prove Conjecture 15 regarding rigidity amplification. This is based on proving a strong parallel repetition theorem for a  $k$ -player game.

Fix some  $k \in \mathbb{N}$ , a function  $f : \{0, 1\}^k \rightarrow \{0, 1\}^k$ , an integer  $1 \leq s < k$ , and  $\mathcal{S} = (S_1, \dots, S_k)$ , where each  $S_i \subseteq [k]$  is of size  $s$ . We define a  $k$ -player game  $\mathcal{G}_{\mathcal{S}}$  as follows:

The  $k$ -players choose functions  $g_1, \dots, g_k : \{0, 1\}^s \rightarrow \{0, 1\}$ , which we call a strategy. A verifier chooses  $x_1, \dots, x_k \in \{0, 1\}^s$  uniformly and independently. Let  $x = (x_1, \dots, x_k) \in \{0, 1\}^{ks}$ . For each  $j \in [k]$ , Player  $j$  is given the input  $x|_{S_j}$ , and they answer  $a_j = g_j(x|_{S_j}) \in \{0, 1\}$ . The verifier accepts if and only if  $f(x) = (a_1, \dots, a_k)$ . The goal of the players is to maximize the winning probability. Formally, the value of the game is defined as

$$\text{val}(\mathcal{G}_{\mathcal{S}}) := \max_{g_1, \dots, g_k} \Pr_{x_1, \dots, x_k \in \{0, 1\}^s} [f(x) = (g_1(x|_{S_1}), \dots, g_k(x|_{S_k}))].$$

The  $n$ -fold repetition of  $\mathcal{G}_{\mathcal{S}}$ , denoted by  $\mathcal{G}_{\mathcal{S}}^{\otimes n}$  is defined as follows. The players choose a strategy  $g_1, \dots, g_k : \{0, 1\}^s \rightarrow \{0, 1\}$ . The verifier chooses  $x_1, \dots, x_k \in \{0, 1\}^s$  uniformly and independently. Let  $x = (x_1, \dots, x_k) \in \{0, 1\}^{ks}$ . Player  $j$  is given the input  $x|_{S_j}$ , and they answer  $a_j = g_j(x|_{S_j}) \in \{0, 1\}$ . The verifier accepts if and only if  $f^{\otimes n}(x) = (a_1, \dots, a_k)$ . That is, for each  $i \in [n]$ ,  $j \in [k]$ , the  $j^{\text{th}}$  bit of  $f(x_{i1}, \dots, x_{ik})$  equals the  $i^{\text{th}}$  bit of  $a_j$ . The value of this repeated game is

$$\text{val}(\mathcal{G}_{\mathcal{S}}^{\otimes n}) := \max_{g_1, \dots, g_k} \Pr_{x_1, \dots, x_k \in \{0, 1\}^s} [f^{\otimes n}(x) = (g_1(x|_{S_1}), \dots, g_k(x|_{S_k}))].$$

From Definition 11, we get the following:

► **Observation 16.** Let  $f : \{0, 1\}^k \rightarrow \{0, 1\}^k$  be a function, and  $n \in \mathbb{N}$ . Then,  $f^{\otimes n}$  is an  $(r, s)$ -block-rigid function if and only if for every  $\mathcal{S} = (S_1, \dots, S_k)$  with set sizes as  $s$ ,  $\text{val}(\mathcal{G}_{\mathcal{S}}^{\otimes n}) < 2^{-r}$ .

**Proof.** Let  $f^{\otimes n}$  be an  $(r, s)$ -block-rigid function. Suppose, for the sake of contradiction, that  $\mathcal{S} = (S_1, \dots, S_k)$  is such that  $\text{val}(\mathcal{G}_{\mathcal{S}}^{\otimes n}) \geq 2^{-r}$ . Let the functions  $g_1, \dots, g_k : \{0, 1\}^s \rightarrow \{0, 1\}$  be an optimal strategy for the players. Define  $X :=$



$\left\{x \in \{0,1\}^{nk} : f^{\otimes n}(x) = (g_1(x|_{S_1}), \dots, g_k(x|_{S_k}))\right\}$ . Then,  $|X| = \text{val}(\mathcal{G}_S) \cdot 2^{nk} \geq 2^{nk-r}$ , which contradicts the block rigidity of  $f^{\otimes n}$ .

Conversely, suppose that  $f^{\otimes n}$  is not  $(r, s)$ -block-rigid. Then, there exists  $X \subseteq \{0,1\}^{nk}$  with  $|X| \geq 2^{nk-r}$ , subsets  $S_1, \dots, S_k \subseteq [k]$  of size  $s$ , and functions  $g_1, \dots, g_k : \{0,1\}^{ns} \rightarrow \{0,1\}^n$ , such that for all  $x \in X$ ,  $f^{\otimes n}(x) = (g_1(x|_{S_1}), \dots, g_k(x|_{S_k}))$ . Let  $\mathcal{S} = (S_1, \dots, S_k)$ , and suppose the players use strategy  $g_1, \dots, g_k$ . Then,  $\text{val}(\mathcal{G}_S^{\otimes n}) \geq |X| \cdot 2^{-nk} \geq 2^{-r}$ .  $\blacktriangleleft$

In particular, for  $n = 1$ , Observation 16 gives the following:

► **Observation 17.** *A function  $f : \{0,1\}^k \rightarrow \{0,1\}^k$  is an  $(r, s)$ -rigid-function if and only if for every  $\mathcal{S} = (S_1, \dots, S_k)$  with set sizes as  $s$ ,  $\text{val}(\mathcal{G}_S) < 2^{-r}$ .*

We conjecture the following strong parallel repetition theorem.

► **Conjecture 18.** *There exists a constant  $c > 0$  such that the following is true. Let  $f : \{0,1\}^k \rightarrow \{0,1\}^k$  be any function, and  $\mathcal{S} = (S_1, \dots, S_k)$  be such that for each  $i \in [k]$ ,  $S_i \subseteq [k]$  is of size  $s$ . Then, for all  $n \in \mathbb{N}$ ,  $\text{val}(\mathcal{G}_S^{\otimes n}) \leq (\text{val}(\mathcal{G}_S))^{cn}$ .*

Combining Observation 16 and 17, we get the following:

► **Theorem 19.** *Conjecture 18  $\implies$  Conjecture 15.*

► **Remarks.**

- (i) *By looking only at some particular player, it can be shown that if  $\text{val}(\mathcal{G}_S) < 1$ , then  $\text{val}(\mathcal{G}_S^{\otimes n}) \leq 2^{-\Omega(n)}$ . In fact, such a result holds for all independent games. The harder part seems to be showing strong parallel repetition when the initial game has small value.*
- (ii) *Observe that the game  $\mathcal{G}_S$  has a randomized predicate in the case  $\cup_{j=1}^k S_j \neq [k]$ . This condition can be removed (even for general independent games) by introducing a new player. This player is given the random string used by the verifier, and is always required to answer a single bit equal to zero. This maintains the independent game property, and ensures that the predicate used by the verifier is a deterministic function of the vector of input queries to the players.*

## 5 Turing Machine Lower Bounds

In this section, we show a conditional super-linear lower bound for multi-tape deterministic Turing machines that can take advice.

Without loss of generality, we only consider machines that have a separate read-only input tape. We assume that the advice string, which is a function of the input length, is written on a separate advice tape at the beginning of computation. We are interested in machines that compute multi-output functions. For this, we assume that at the end of computation, the machine writes the entire output on a separate write-only output tape, and then halts.

We consider the following problem.

- **Definition 20.** *Let  $k : \mathbb{N} \rightarrow \mathbb{N}$  be a function. We define the problem  $\text{Tensor}_k$  as follows:*
- *Input:  $(f, x)$ , where  $f : \{0,1\}^k \rightarrow \{0,1\}^k$  is a function, and  $x \in \{0,1\}^{nk}$ , for some  $n \in \mathbb{N}$  and  $k = k(n)$ .*
  - *Output:  $f^{\otimes n}(x) \in \{0,1\}^{nk}$ .*

*The function  $f$  is given as input in the form of its entire truth table. The input  $x = (x_{ij})_{i \in [n], j \in [k]}$  is given in the order  $(x_{11}, \dots, x_{n1}, x_{12}, \dots, x_{n2}, \dots, x_{1k}, \dots, x_{nk})$ . The total length of the input is  $m(n) := 2^k k + nk$ .*

We observe that if the function  $k : \mathbb{N} \rightarrow \mathbb{N}$  grows very slowly with  $n$ , the problem  $\text{Tensor}_k$  can be solved by a deterministic Turing machine in slightly super-linear time.

► **Observation 21.** *Let  $k : \mathbb{N} \rightarrow \mathbb{N}$  be a function. There exists a deterministic Turing machine that solves the problem  $\text{Tensor}_k$  in time  $O(nk2^k)$ , on input  $(f, x)$  of length  $nk + 2^k k$ , where  $k = k(n)$ .*

**Proof.** We note that applying  $f^{\otimes n}$  on  $x = (x_{ij})_{i \in [n], j \in [k]}$  is the same as applying  $f$  on  $(x_{i1}, x_{i2}, \dots, x_{ik})$ , in place for each  $i \in [n]$ . A Turing machine can do the following:

1. Find  $n$  and  $k$ , using the fact that the description of  $f$  is of length  $2^k k$ , and that of  $x$  is of length  $nk$ .
2. Rearrange the input so that for each  $i \in [n]$ , the part  $(x_{i1}, x_{i2}, \dots, x_{ik})$  is written consecutively on the tape.
3. Using the truth table of  $f$ , compute the output for each such part.
4. Rearrange the entire output back to the desired form.

The total time taken is  $O(nk + nk^2 + nk2^k + nk^2) = O(nk2^k)$ . ◀

We now state and prove the main technical theorem of this section.

► **Theorem 22.** *Let  $k : \mathbb{N} \rightarrow \mathbb{N}$  be a function such that  $k(n) = \omega(1)$  and  $k(n) = o(\log_2 n)$ . Let  $m = m(n) := 2^k k + nk$ , where  $k = k(n)$ .*

*Suppose  $M$  is a deterministic multi-tape Turing machine that takes advice, and runs in linear time in the length of its input. Assuming Conjecture 15, the machine  $M$  does not solve the problem  $\text{Tensor}_k$  correctly for all inputs. That is, there exists  $n \in \mathbb{N}$ , and  $y = (f, x) \in \{0, 1\}^m$  such that  $M(y) \neq f^{\otimes n}(x)$ .*

There are two main technical ideas that will be useful to us. The first is the notion of block-respecting Turing machines, defined by Hopcroft, Paul and Valiant [20]. The second is a graph theoretic result, which was proven by Paul, Pippenger, Szemerédi and Trotter [28], and was used to show a separation between deterministic and non-deterministic linear time.

► **Definition 23.** *Let  $M$  be a Turing machine, and let  $b : \mathbb{N} \rightarrow \mathbb{N}$  be a function. Partition the computation of  $M$ , on any input  $y$  of length  $m$ , into time segments of length  $b(m)$ , with the last segment having length at most  $b(m)$ . Also, partition each of the tapes of  $M$  into blocks, each consisting of  $b(m)$  contiguous cells.*

*We say that  $M$  is block-respecting with respect to block size  $b$ , if on inputs of length  $m$ , the tape heads of  $M$  cross blocks only at times that are integer multiples of  $b(m)$ .*

► **Lemma 24.** [20] *Let  $t : \mathbb{N} \rightarrow \mathbb{N}$  be a function, and  $M$  be a multi-tape deterministic Turing machine running in time  $t(m)$  on inputs of length  $m$ . Let  $b : \mathbb{N} \rightarrow \mathbb{N}$  be a function such that  $b(m)$  is computable from  $1^m$  by a multi-tape deterministic Turing machine running in time  $O(t(m))$ . Then, the language recognized by  $M$  is also recognized by a multi-tape deterministic Turing Machine  $M'$ , which runs in time  $O(t(m))$ , and is block-respecting with respect to  $b$ .*

The rest of this section is devoted to the proof of Theorem 22. Let  $k : \mathbb{N} \rightarrow \mathbb{N}$  be a function such that  $k(n) = \omega(1)$  and  $k(n) = o(\log_2 n)$ . Let  $m = m(n) := 2^k k + nk$ , where  $k = k(n)$ .

Suppose that  $M$  is a deterministic multi-tape Turing Machine, which on input  $y = (f, x) \in \{0, 1\}^m$ , takes advice, runs in time  $O(m)$ , and outputs  $f^{\otimes n}(x)$ .

Let  $b : \mathbb{N} \rightarrow \mathbb{N}$  be a function such that  $b(m) = n$ . By our assumption that  $k(n) = o(\log_2 n)$ , we can assume that the input  $y = (f, x) \in \{0, 1\}^m$  consists of  $k + 1$  blocks, where the first block contains  $f$  (possibly padded by blank symbols to the left), and the remaining  $k$  blocks

contain  $x$ . By Lemma 24, we can further assume that  $M$  is block-respecting with respect to  $b$ . Note that we can assume  $n$  to be a part of the advice, and hence we don't need to care about the computability of  $b$ .

For an input  $y = (f, x) \in \{0, 1\}^m$ , the number of time segments for which  $M$  runs on  $x$  is at most  $\frac{O(m)}{b(m)} = \frac{O(nk)}{n} = O(k) := a(n)$ .

We define the computation graph  $G_M(y)$ , for input  $y \in \{0, 1\}^m$ , as follows.

► **Definition 25.** *The vertex set of  $G_M(y)$  is defined to be  $V_M(y) = \{1, \dots, a(n)\}$ . For each  $1 \leq i < j \leq a(n)$ , the edge set  $E_M(y)$  has the edge  $(i, j)$ , if either*

- (i)  $j = i + 1$ , or
- (ii) *there is some tape, such that, during the computation on  $y$ ,  $M$  visits the same block on that tape in both time-segments  $i$  and  $j$ , and never visits that block during any time-segment strictly between  $i$  and  $j$ .*

In a directed acyclic graph  $G$ , we say that a vertex  $u$  is a predecessor of a vertex  $v$ , if there exists a directed path from  $u$  to  $v$ .

► **Lemma 26** ([28]). *For every  $y$ , the graph  $G_M(y)$  satisfies the following:*

1. *Each vertex in  $G_M(y)$  has degree  $O(1)$ .*
2. *There exists a set of vertices  $J \subset V_M(y)$  in  $G_M(y)$ , of size  $O\left(\frac{a(n)}{\log^* a(n)}\right)$  such that every vertex of  $G_M(y)$  has at most  $O\left(\frac{a(n)}{\log^* a(n)}\right)$  many predecessors in the induced subgraph on the vertex set  $V_M(y) \setminus J$ .*

*We note that the constants here might depend on the number of tapes of  $M$ .*

► **Lemma 27.** *Let  $\epsilon > 0$  be any constant and  $\mathcal{Y} \subseteq \{0, 1\}^m$  be any subset of the inputs. For all (large enough)  $n$ , there exists a subset  $Y \subseteq \mathcal{Y}$  of size  $|Y| \geq |\mathcal{Y}| \cdot 2^{-\epsilon nk}$ , and subsets  $S_1, \dots, S_k \subseteq [k]$  of size  $\epsilon k$ , such that for each  $y = (f, x) \in Y$ , and each  $i \in [k]$ , the  $i^{\text{th}}$  block (of length  $n$ ) of  $f^{\otimes n}(x)$  can be written as a function of  $x|_{S_i}$  and the truth-table of  $f$ .*

**Proof.** For input  $y = (f, x) \in \{0, 1\}^m$ , let  $J(y) \subset V_M(y)$  be a set as in Lemma 26.

Let  $C(y)$  denote the following information about the computation of  $M$ :

- (i) The internal state of  $M$  at the end of each time-segment.
- (ii) The position of all tape heads at the end of each time-segment.
- (iii) For each time segment in  $J(y)$ , and for each tape of  $M$ , the final transcription (of length  $n$ ) of the block that was visited on this tape during this segment.

Let  $g : \mathcal{Y} \rightarrow \{0, 1\}^*$  be the function given by  $g(y) = (G_M(y), J(y), C(y))$ . Observe that the output of  $g$  can be described using  $O\left(k \log_2 k + \frac{nk}{\log^* k}\right)$  bits. By our assumption that  $k(n) = \omega(1)$  and  $k(n) = o(\log_2 n)$ , we have that for large  $n$ , this is at most  $\epsilon nk$  bits. Hence, there exists a set  $Y \subseteq \mathcal{Y}$  of size  $|Y| \geq |\mathcal{Y}| \cdot 2^{-\epsilon nk}$ , such that for each  $y \in Y$ ,  $g(y)$  takes on some fixed value  $(G = (V, E), J, C)$ .

Now, consider any  $y = (f, x) \in Y$ . The machine writes the  $k$  blocks of the output  $f^{\otimes n}(x)$  on the output tape in the last  $k$  time segments before halting. For each of these time segments, the corresponding vertex in  $G$  has at most  $O\left(\frac{k}{\log^* k}\right) \leq \epsilon k$  predecessors in the induced subgraph on  $V \setminus J$ . These further correspond to at most  $\epsilon k$  distinct blocks of  $y$  that are visited (on the input tape) during these predecessor time segments. Since the relevant block transcriptions at the end of time segments for vertices in  $J$  are fixed in  $C$ , each output block can be written as a function of at most  $\epsilon k$  blocks of  $y$ . For the  $i^{\text{th}}$  block of output, without loss of generality, this includes the first block of  $y$ , which contains the truth table of  $f$ , and blocks of  $x$  which indexed by some subset  $S_i \subseteq [k]$  of size  $\epsilon k$ . ◀

**Proof of Theorem 22.** Let  $\delta = \frac{1}{8}$ . Fix some sufficiently large  $n$ , and a  $(\delta k, \delta k)$ -rigid function  $f_0 : \{0, 1\}^k \rightarrow \{0, 1\}^k$ . The existence of such a function is guaranteed by Proposition 8. By Conjecture 15, the function  $f_0^{\otimes n}$  is an  $(\epsilon n k, \epsilon k)$ -block-rigid function for some constant  $\epsilon > 0$ . On the other hand, Lemma 27, with  $\mathcal{Y} = \{(f_0, x) : x \in \{0, 1\}^{nk}\}$ , shows that  $f_0^{\otimes n}$  is not an  $(\epsilon n k, \epsilon k)$ -block-rigid function for any constant  $\epsilon > 0$ . ◀

We now restate and prove Theorem 2.

▶ **Theorem 28.** *Let  $t : \mathbb{N} \rightarrow \mathbb{N}$  be any function such that  $t(n) = \omega(n)$ . Assuming Conjecture 15, there exists a function  $f : \{0, 1\}^* \rightarrow \{0, 1\}^*$  such that*

1. *On inputs  $x$  of length  $n$  bits, the output  $f(x)$  is of length at most  $n$  bits.*
2. *The function  $f$  is computable by a multi-tape deterministic Turing machine that runs in time  $O(t(n))$  on inputs of length  $n$ .*
3. *The function  $f$  is not computable by any multi-tape deterministic Turing machine that takes advice and runs in time  $O(n)$  on inputs of length  $n$ .*

**Proof.** Let  $k : \mathbb{N} \rightarrow \mathbb{N}$  be a function such that  $k(n) = \omega(1)$ ,  $k(n) = o(\log_2 n)$ , and  $nk2^k \leq t(2^k k + nk)$ . The theorem then follows from Observation 21 and Theorem 22. ◀

▶ **Remarks.**

- (i) *We note that for the proof of Theorem 22, it suffices to find, for infinitely many  $n$ , a single function  $f : \{0, 1\}^{k(n)} \rightarrow \{0, 1\}^{k(n)}$  such that  $f^{\otimes n}$  is an  $(\epsilon n k, \epsilon k)$ -block-rigid function, where  $\epsilon > 0$  is a constant. This would show that  $M$  cannot give the correct answer to  $\text{Tensor}_k$  for inputs of the form  $(f, x)$ , where  $x \in \{0, 1\}^{nk}$ .*
- (ii) *For the proof of Theorem 22, it can be shown that it suffices for the following condition to hold for infinitely many  $n$ , and some constant  $\epsilon > 0$ . Let  $S_1, \dots, S_k \subseteq [k]$  be fixed sets of size  $\epsilon k$ , and  $f : \{0, 1\}^k \rightarrow \{0, 1\}^k$  be a function chosen uniformly at random. Then, with probability at least  $1 - 2^{-\omega(k \log_2 k)}$ , the function  $f^{\otimes n}$  is an  $(\epsilon n k, \epsilon k)$ -block-rigid function against the fixed sets  $S_1, \dots, S_k$ . We note that the probability here is not good enough to be able to union bound over  $S_1, \dots, S_k$  and get a single function as mentioned in the previous remark.*

Essentially the same argument as that of Theorem 22 also proves Theorem 3, which we restate below.

▶ **Theorem 29.** *Let  $k : \mathbb{N} \rightarrow \mathbb{N}$  be a function such that  $k(n) = \omega(1)$  and  $k(n) = 2^{o(n)}$ , and  $f_n : \{0, 1\}^{nk(n)} \rightarrow \{0, 1\}^{nk(n)}$  be a family of  $(\epsilon n k(n), \epsilon k(n))$ -block-rigid functions, for some constant  $\epsilon > 0$ . Let  $M$  be any multi-tape deterministic linear-time Turing machine that takes advice. Then, there exists  $n \in \mathbb{N}$ , and  $x \in \{0, 1\}^{nk(n)}$ , such that  $M(x) \neq f_n(x)$ .*

The above theorem makes it interesting to find families of block-rigid functions that are computable in polynomial time.

## 6 Size-Depth Tradeoffs

In this section, we will consider boolean circuits over a set  $F$ . These are directed acyclic graphs with each node  $v$  labelled either as an input node or by an arbitrary function  $g_v : F \times F \rightarrow F$ . The input nodes have in-degree 0 and all other nodes have in-degree 2. Some nodes are further labelled as output nodes, and they compute the outputs (in the usual manner), when the inputs are from the set  $F$ . The size of the circuit is defined to be the number of edges in the graph. The depth of the circuit is defined to be the length of a longest directed path from an input node to an output node.

Valiant [36] showed that if  $A \in \mathbb{F}^{n \times n}$  is an  $(\epsilon n, n^\epsilon)$ -rigid matrix for some constant  $\epsilon > 0$ , then the corresponding function cannot be computed by an  $O(n)$ -size and  $O(\log_2 n)$ -depth linear circuit over  $\mathbb{F}$ . By a linear circuit, we mean that each gate computes a linear function (over  $\mathbb{F}$ ) of its inputs. A similar argument can be used to prove the following.

► **Lemma 30** ([36]). *Suppose  $f : \{0, 1\}^{nk} \rightarrow \{0, 1\}^{nk}$  is an  $(\epsilon nk, k^\epsilon)$ -block-rigid function, for some constant  $\epsilon > 0$ . Then, the function  $g : (\{0, 1\}^n)^k \rightarrow (\{0, 1\}^n)^k$  corresponding to  $f$  cannot be computed by an  $O(k)$ -size and  $O(\log_2 k)$ -depth circuit over the set  $F = \{0, 1\}^n$ .*

► **Theorem 31.** *Let  $k : \mathbb{N} \rightarrow \mathbb{N}$  be a function such that  $k(n) = \omega(1)$  and  $k(n) = o(\log_2 n)$ . Let  $m = m(n) := 2^k k + nk$ , where  $k = k(n)$ .*

*Assuming Conjecture 15, the problem  $\text{Tensor}_k$  is not solvable by  $O(k)$ -size and  $O(\log_2 k)$ -depth circuits over the set  $F = \{0, 1\}^n$ . Here, the input  $(f, x)$  to the circuit is given in the form of  $k + 1$  elements in  $\{0, 1\}^n$ , the first one being the truth table of  $f$ , and the remaining  $k$  being the blocks of  $x$ .*

**Proof.** Let  $\delta = \frac{1}{8}$ . Fix some large  $n$ , and a  $(\delta k, \delta k)$ -rigid function  $f_0 : \{0, 1\}^k \rightarrow \{0, 1\}^k$ , where  $k = k(n)$ . The existence of such a function is guaranteed by Proposition 8. Assuming Conjecture 15, the function  $f_0^{\otimes n}$  is an  $(\epsilon nk, \epsilon k)$ -block-rigid function, for some universal constant  $\epsilon > 0$ . By Lemma 30, the corresponding function on  $(\{0, 1\}^n)^k$  cannot be computed by an  $O(k)$ -size and  $O(\log_2 k)$ -depth circuit over  $F = \{0, 1\}^n$ . Since  $f_0$  can be hard-wired in any circuit solving  $\text{Tensor}_k$ , we have the desired result. ◀

## 7 Rigid Matrices and Rigid Functions

A natural question to ask is whether the functions corresponding to rigid matrices are rigid functions or not.

► **Conjecture 32.** *There exists a universal constant  $c > 0$  such that whenever  $A \in \mathbb{F}_2^{n \times n}$  is an  $(r, s)$ -rigid matrix, the corresponding function  $A : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$  is a  $(cr, cs)$ -rigid function.*

We show that a positive answer to the above resolves a closely related conjecture by Jukna and Schnitger [21].

► **Definition 33.** *Consider a depth-2 circuit, with  $x = (x_1, \dots, x_n)$  as the input variables,  $w$  gates in the middle layer, computing boolean functions  $h_1, \dots, h_w$  and  $m$  output gates, computing boolean functions  $g_1, \dots, g_m$ . The circuit computes a function  $f = (f_1, \dots, f_m) : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^m$  satisfying  $f_i(x_1, \dots, x_n) = g_i(x, h_1(x), \dots, h_w(x))$ , for each  $i \in [m]$ . The width of the circuit is defined to be  $w$ . The degree of the circuit is defined to be the maximum over all gates  $g_i$ , of the number of wires going directly from the inputs  $x_1, \dots, x_n$  to  $g_i$ .*

We remark that Lemma 27 essentially shows that any function computable by a deterministic linear-time Turing Machine has a depth-2 circuit of small width and small “block-degree”.

► **Conjecture 34** ([21]). *Suppose  $f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$  is a linear function computable by a depth-2 circuit with width  $w$  and degree  $d$ . Then,  $f$  is computable by a depth-2 circuit, with width  $O(w)$ , and degree  $O(d)$ , each of whose gates compute a linear function.*

► **Observation 35.** *Conjecture 32  $\implies$  Conjecture 34.*

**Proof.** Suppose  $f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$  is a linear function computable by a depth-2 circuit with width  $w$  and degree  $d$ . Then, there exists a set  $X \subseteq \mathbb{F}_2^n$  of size at least  $2^{n-w}$ , such that for each  $x \in X$ , the value of the functions computed by the gates in the middle layer is the same.

Hence, for each  $x \in X$ , each element of  $f(x)$  can be written as a function of at most  $d$  elements of  $x$ . This shows that  $f$  is not an  $(w, d)$ -rigid function. Assuming Conjecture 32, the matrix  $A \in \mathbb{F}_2^{n \times n}$  for the function  $f$  is not a  $(cw, cd)$ -rigid matrix, for some constant  $c > 0$ . Then,  $A = B + C$ , where the rank of  $B$  is at most  $cw$ , and  $C$  has at most  $cd$  non-zero entries in each row. Now, there exist matrices  $B_1 \in \mathbb{F}_2^{n \times cw}$  and  $B_2 \in \mathbb{F}_2^{cw \times n}$ , such that  $B = B_1 B_2$ . Then,  $f$  is computable by a linear depth-2 circuit with width  $cw$  and degree  $cd$ , where the middle layer computes output of the function corresponding to  $B_2$ . ◀

## 8 Future Directions

In this section, we state some well known problems related to matrices. It seems interesting to study the block-rigidity of these functions.

### 8.1 Matrix Transpose

The matrix-transpose problem is described as follows:

- *Input:* A matrix  $X \in \mathbb{F}_2^{n \times n}$  as a vector of length  $n^2$  bits, in row-major order, for some  $n \in \mathbb{N}$ .
- *Output:* The matrix  $X$  column-major order (or equivalently, the transpose of  $X$  in row-major order).

It is well known (see [6] for a short proof) that the above problem can be solved on a 2-tape Turing machine in time  $O(N \log N)$ , on inputs of length  $N = n^2$ . We believe that this cannot be solved by Turing machines in linear-time, and that the notion of block-rigidity might be a viable approach to prove this. Next, we observe some structural details about the problem.

The matrix-transpose problem computes a linear function, whose  $N \times N$  matrix  $A$  on inputs of length  $N = n^2$  is described as follows. For each  $i, j \in [n]$ , let  $e_{ij} \in \mathbb{F}_2^{n \times n}$  denote the matrix whose  $(i, j)$ <sup>th</sup> entry is 1 and rest of the entries are zero. The matrix  $A$  is an  $N \times N$  matrix made up of  $n^2$  blocks, with the  $(i, j)$ <sup>th</sup> block equal to  $e_{ji}$ .

Using a similar argument as in Observation 16, one can show that the value of the following game captures the block-rigidity of the matrix-transpose function. Fix integers  $n \in \mathbb{N}$ ,  $1 \leq s < n$ , and a collection  $\mathcal{S} = (S_1, \dots, S_n)$ , where each  $S_i \subseteq [n]$  is of size  $s$ . We define an  $n$ -player game  $\mathcal{G}_{\mathcal{S}}$  as follows: A verifier chooses a matrix  $X \in \mathbb{F}_2^{n \times n}$ , with each entry being chosen uniformly and independently. For each  $j \in [n]$ , player  $j$  is given the rows of the matrix indexed by  $S_j$ , and they answer  $y_j \in \mathbb{F}_2^n$ . The verifier accepts if and only if for each  $j \in [n]$ ,  $y_j$  equals the  $j$ <sup>th</sup> column of  $X$ .

► **Conjecture 36.** *There exists a constant  $c > 0$  such that the function given by the matrix  $A$  is a  $(cn^2, cn)$ -block-rigid function. Equivalently, for each collection  $\mathcal{S}$  with set sizes as  $cn$ , the value of the game  $\mathcal{G}_{\mathcal{S}}$  is at most  $2^{-cn^2}$ .*

We note that the above game is of independent interest from a combinatorial point of view as well. Basically, it asks whether there exists a large family of  $n \times n$  matrices, in which each column can be represented as some function of a small fraction of the rows. The problem of whether the matrix  $A$  is a block-rigid matrix is also interesting. This corresponds to the players in the above game using strategies which are linear functions.



## 8.2 Matrix Product

The matrix-product problem is described as follows:

- *Input:* Matrices  $X, Y \in \mathbb{F}_2^{n \times n}$  as vectors of length  $n^2$  bits, in row-major order, for some  $n \in \mathbb{N}$ .
- *Output:* The matrix  $Z = XY$  in row-major order.

The block-rigidity of the matrix-product function is captured by the following game: Fix integers  $n \in \mathbb{N}$ ,  $1 \leq s < n$ , and collections  $\mathcal{S} = (S_1, \dots, S_n)$ ,  $\mathcal{T} = (T_1, \dots, T_n)$  where each  $S_i, T_i \subseteq [n]$  is of size  $s$ . We define a  $n$ -player game  $\mathcal{G}_{\mathcal{S}, \mathcal{T}}$  as follows: A verifier chooses matrices  $X, Y \in \mathbb{F}_2^{n \times n}$ , with each entry being chosen uniformly and independently. For each  $j \in [n]$ , player  $j$  is given the rows of the matrices  $X$  and  $Y$  indexed by  $S_j$  and  $T_j$  respectively, and they answer  $y_j \in \mathbb{F}_2^n$ . The verifier accepts if and only if for each  $j \in [n]$ ,  $y_j$  equals the  $j^{\text{th}}$  row of  $XY$ .

► **Conjecture 37.** *There exists a constant  $c > 0$  such that for each  $\mathcal{S}, \mathcal{T}$  with set sizes as  $cn$ , the value of the game  $\mathcal{G}_{\mathcal{S}, \mathcal{T}}$  is at most  $2^{-cn^2}$ .*

One may change the row-major order for some (or all) of the matrices to column-major order. It is easy to modify the above game in such a case.

---

## References

- 1 Josh Alman and Lijie Chen. Efficient construction of rigid matrices using an NP oracle. In *FOCS*, pages 1034–1055, 2019.
- 2 Josh Alman and Ryan Williams. Probabilistic rank and matrix rigidity. In *STOC*, pages 641–652, 2017.
- 3 Michael Ben-Or, Shafi Goldwasser, Joe Kilian, and Avi Wigderson. Multi-prover interactive proofs: How to remove intractability assumptions. In *STOC*, pages 113–131, 1988.
- 4 Amey Bhangale, Prahladh Harsha, Orr Paradise, and Avishay Tal. Rigid matrices from rectangular PCPs. In *FOCS*, 2020. accepted.
- 5 Mark Braverman and Ankit Garg. Small value parallel repetition for general games. In *STOC*, pages 335–340, 2015.
- 6 Martin Dietzfelbinger, Wolfgang Maass, and Georg Schnitger. The complexity of matrix transposition on one-tape off-line Turing machines. *Theoret. Comput. Sci.*, 82:113–129, 1991.
- 7 Irit Dinur, Prahladh Harsha, Rakesh Venkat, and Henry Yuen. Multiplayer parallel repetition for expanding games. In *ITCS*, volume 67 of *LIPICs*, pages Art. No. 37, 16, 2017.
- 8 Irit Dinur and David Steurer. Analytical approach to parallel repetition. In *STOC*, pages 624–633, 2014.
- 9 Zeev Dvir, Alexander Golovnev, and Omri Weinstein. Static data structure lower bounds imply rigidity. In *STOC*, pages 967–978, 2019.
- 10 Uriel Feige. On the success probability of the two provers in one-round proof systems. In *Structure in Complexity Theory Conference*, pages 116–123, 1991.
- 11 Uriel Feige and Oleg Verbitsky. Error reduction by parallel repetition – a negative result. *Combinatorica*, 22(4):461–478, 2002.
- 12 Lance Fortnow, Richard Lipton, Dieter van Melkebeek, and Anastasios Viglas. Time-space lower bounds for satisfiability. *J. ACM*, 52(6):835–865, 2005.
- 13 Lance Fortnow, John Rempel, and Michael Sipser. On the power of multi-prover interactive protocols. *Theoret. Comput. Sci.*, 134(2):545–557, 1994.
- 14 Lance Jeremy Fortnow. *Complexity theoretic aspects of interactive proof systems*. PhD thesis, MIT, 1989.
- 15 Joel Friedman. A note on matrix rigidity. *Combinatorica*, 13(2):235–239, 1993.



- 16 H. Furstenberg and Y. Katznelson. A density version of the Hales-Jewett theorem. *J. Anal. Math.*, 57:64–119, 1991.
- 17 Oded Goldreich and Avishay Tal. Matrix rigidity of random Toeplitz matrices. *Comput. Complexity*, 27(2):305–350, 2018. (also in STOC 2016).
- 18 J. Hartmanis and R. E. Stearns. On the computational complexity of algorithms. *Trans. Amer. Math. Soc.*, 117:285–306, 1965.
- 19 Thomas Holenstein. Parallel repetition: simplifications and the no-signaling case. *Theory Comput.*, 5:141–172, 2009. (also in STOC 2007).
- 20 John Hopcroft, Wolfgang Paul, and Leslie Valiant. On time versus space. *J. Assoc. Comput. Mach.*, 24(2):332–337, 1977. (also in FOCS 1975).
- 21 Stasys Jukna and Georg Schnitger. Min-rank conjecture for log-depth circuits. *J. Comput. System Sci.*, 77(6):1023–1038, 2011.
- 22 Kiran S. Kedlaya and Christopher Umans. Fast polynomial factorization and modular composition. *SIAM J. Comput.*, 40(6):1767–1802, 2011. (also in STOC 2008).
- 23 Abhinav Kumar, Satyanarayana V. Lokam, Vijay M. Patankar, and M. N. Jayalal Sarma. Using elimination theory to construct rigid matrices. *Comput. Complexity*, 23(4):531–563, 2014. (also in FSTTCS 2009).
- 24 Satyanarayana V. Lokam. On the rigidity of Vandermonde matrices. *Theoret. Comput. Sci.*, 237(1-2):477–483, 2000.
- 25 Satyanarayana V. Lokam. Spectral methods for matrix rigidity with applications to size-depth trade-offs and communication complexity. *J. Comput. System Sci.*, 63(3):449–473, 2001.
- 26 Satyanarayana V. Lokam. Quadratic lower bounds on matrix rigidity. In *Theory and Applications of Models of Computation*, pages 295–307, 2006.
- 27 Wolfgang Paul and Rüdiger Reischuk. On alternation. II. A graph-theoretic approach to determinism versus nondeterminism. *Acta Inform.*, 14(4):391–403, 1980.
- 28 Wolfgang J. Paul, Nicholas Pippenger, Endre Szemerédi, and William T. Trotter. On determinism versus non-determinism and related problems. In *FOCS*, pages 429–438, 1983.
- 29 D. H. J. Polymath. A new proof of the density Hales-Jewett theorem. *Ann. of Math. (2)*, 175(3):1283–1327, 2012.
- 30 Ran Raz. A parallel repetition theorem. *SIAM J. Comput.*, 27(3):763–803, 1998. (also in STOC 1995).
- 31 Ran Raz. A counterexample to strong parallel repetition. *SIAM J. Comput.*, 40(3):771–777, 2011. (also in FOCS 2008).
- 32 Alexander Razborov. On rigid matrices (in russian). Unpublished Manuscript, 1989.
- 33 Rahul Santhanam. On separators, segregators and time versus space. In *CCC*, pages 286–294, 2001.
- 34 M. A. Shokrollahi, D. A. Spielman, and V. Stemmann. A remark on matrix rigidity. *Inform. Process. Lett.*, 64(6):283–285, 1997.
- 35 Victor Shoup and Roman Smolensky. Lower bounds for polynomial evaluation and interpolation problems. *Comput. Complexity*, 6(4):301–311, 1996/97.
- 36 Leslie G. Valiant. Graph-theoretic arguments in low-level complexity. In *MFCS*, volume 53 of *Lecture Notes in Computer Science*, pages 162–176, 1977.
- 37 Dieter van Melkebeek and Ran Raz. A time lower bound for satisfiability. *Theoret. Comput. Sci.*, 348(2-3):311–320, 2005. (also in ICALP 2004).
- 38 Oleg Verbitsky. Towards the parallel repetition conjecture. *Theoret. Comput. Sci.*, 157(2):277–282, 1996.
- 39 Henning Wunderlich. On a theorem of Razborov. *Comput. Complexity*, 21(3):431–477, 2012.



# Lower Bounds for Off-Chain Protocols: Exploring the Limits of Plasma

**Stefan Dziembowski**

University of Warsaw, Poland

Stefan.Dziembowski@crypto.edu.pl

**Grzegorz Fabiański**

University of Warsaw, Poland

Grzegorz.Fabianski@crypto.edu.pl

**Sebastian Faust**

Technische Universität Darmstadt, Germany

sebastian.faust@tu-darmstadt.de

**Siavash Riahi**

Technische Universität Darmstadt, Germany

siavash.riahi@tu-darmstadt.de

---

## Abstract

Blockchain is a disruptive new technology introduced around a decade ago. It can be viewed as a method for recording timestamped transactions in a public database. Most of blockchain protocols do not scale well, i.e., they cannot process quickly large amounts of transactions. A natural idea to deal with this problem is to use the blockchain only as a timestamping service, i.e., to hash several transactions  $tx_1, \dots, tx_m$  into one short string, and just put this string on the blockchain, while at the same time posting the hashed transactions  $tx_1, \dots, tx_m$  to some public place on the Internet (“off-chain”). In this way the transactions  $tx_i$  remain timestamped, but the amount of data put on the blockchain is greatly reduced. This idea was introduced in 2017 under the name *Plasma* by Poon and Buterin. Shortly after this proposal, several variants of Plasma have been proposed. They are typically built on top of the Ethereum blockchain, as they strongly rely on so-called *smart contracts* (in order to resolve disputes between the users if some of them start cheating). Plasmas are an example of so-called *off-chain protocols*.

In this work we initiate the study of the inherent limitations of Plasma protocols. More concretely, we show that in every Plasma system the adversary can either (a) force the honest parties to communicate a lot with the blockchain, even though they did not intend to (this is traditionally called *mass exit*); or (b) an honest party that wants to leave the system needs to quickly communicate large amounts of data to the blockchain. What makes these attacks particularly hard to handle in real life is that these attacks do not have so-called *uniquely attributable faults*, i.e. the smart contract cannot determine which party is malicious, and hence cannot force it to pay the fees for the blockchain interaction. An important implication of our result is that the benefits of two of the most prominent Plasma types, called *Plasma Cash* and *Fungible Plasma*, cannot be achieved simultaneously.

Besides of the direct implications on real-life cryptocurrency research, we believe that this work may open up a new line of theoretical research, as, up to our knowledge, this is the first work that provides an impossibility result in the area of off-chain protocols.

**2012 ACM Subject Classification** Security and privacy; Security and privacy → Cryptography

**Keywords and phrases** blockchain, lower bounds, off-chain protocol, commit chain, plasma

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.72

**Related Version** A full version of the paper is available at <https://eprint.iacr.org/2020/175>.

**Funding** This work was partly supported by the FY18-0023 PERUN grant from the Ethereum Foundation, by the TEAM/2016-1/4 grant from the Foundation for Polish Science, by the DFG CRC 1119 CROSSING (project S7), by the German Federal Ministry of Education and Research (BMBF)



© Stefan Dziembowski, Grzegorz Fabiański, Sebastian Faust, and Siavash Riahi;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 72; pp. 72:1–72:20



Leibniz International Proceedings in Informatics

LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

iBlockchain project (grant nr. 16KIS0902), and by the German Federal Ministry of Education and Research and the Hessen State Ministry for Higher Education, Research and the Arts within their joint support of the National Research Center for Applied Cybersecurity ATHENE.

## 1 Introduction

What does it mean to *timestamp* a digital document? Haber and Stornetta in their seminal paper [17] define timestamping as a method to certify when a given document was created. In many settings the timestamped document  $T$  remains secret after it was timestamped, until its creator decides to make it public. This is often because of efficiency reasons – for example in the scheme of [17] what is really timestamped is the cryptographic hash<sup>1</sup>  $H(T)$  (not  $T$ ), which leads to savings in communication between  $T$ 's creator and the timestamping service. Sometimes the secrecy of  $T$  is actually a desired feature. According to [29] several researchers in the past (including Galileo Galilei and Isaac Newton) have used ad-hoc methods to timestamp their research ideas before publishing them, in order to later claim priority.

Recently, in the context of blockchain, timestamping has been used in a slightly different way. Namely, in the paper that introduced this technology [25], the “timestamping” mechanism is such that  $T$ 's creator does not only get a proof that  $T$  has been created at a given time, but also that it has been *made public* at this time. Let us call this kind of scheme *public timestamping*, and let us refer to the former type of timestamping as *secret timestamping*. The public timestamping feature of blockchain has been one of the main reasons why this technology attracted so much attention. In fact one of the first projects that use blockchain for purposes other than purely financial was *Namecoin* that used the timestamping to create a decentralized domain name system (see, e.g., [19] for more on this project).

In many cases timestamping is expensive, and its costs grow linearly with the length of the timestamped document. This is especially true for blockchain-based solutions, where all parties in the network need to reach *consensus* about what document was published and when. For example, Bitcoin (the blockchain system introduced in [25]) can process at most around 1MB of data per 10 minutes. In Ethereum, which is another very popular blockchain system [31] “timestamping” a word of 32 bytes cost currently around USD 0.80. A similar problem appears in several other blockchain protocols. We give a short introduction to blockchain in Sec. 1.1. For a moment let us just say that typical blockchains come with their own virtual currencies (also called *cryptocurrencies*). In the most standard case the “timestamped” messages define financial transfers between the network participants, and are hence called *transactions*. We will refer to timestamping a transaction as *posting it on the blockchain*.

To summarize, from the efficiency point of view the secret timestamping is better than the public one (as only hashes need to be timestamped). From the security perspective these two types of timestamping are incomparable as the fact that the timestamped document  $T$  needs to be published can be considered to be both an advantage and a disadvantage, depending on the particular application. For example, one could argue that considering timestamped hashes of academic papers as being sufficient evidence for claiming priority would slow down scientific progress, as it would disincentive making the papers public (until, say, the author writes down all followup papers that build upon the timestamped paper).

---

<sup>1</sup> In this paper we assume reader's familiarity with basic cryptographic notions such as hash functions, negligible functions, etc. For an introduction to this topic see, e.g., [20].

A natural question therefore is as follows. Suppose we have access to a timestamping service that is expensive to use (so we would rather timestamp only very short messages there). Can we use it to “emulate” public timestamping that would be cheap for long documents  $T$ ? Of course this “emulation” cannot work in general, since some scenarios may simply require a proof that the whole  $T$  was available publicly at a given time. So the best we can hope for is to find *some* applications which permit such emulation. An idea for such emulation emerged recently within the cryptocurrency community under the names *Plasma* or *commit chains*<sup>2</sup> [26, 21, 16, 28, 24]. Very informally speaking, Plasma allows to “compress” a number of blockchain transactions  $tx_1, \dots, tx_m$  into one very short string  $h = H(T)$ , where  $T := (tx_1, \dots, tx_m)$ . The transactions can come from different parties called *users*. The only document that gets posted on the blockchain is  $h$ . The compression and posting  $h$  is done by one designated party called the *operator* (denoted  $Op$ ). Besides of posting  $h$ , the operator publishes all the transactions  $T$  on some public network (say: on her web-page). Since this publication is *not* done on the blockchain we also say that it is performed *off-chain*.

Publishing  $T$  off-chain is important since only then each user can verify that her transaction was indeed included into the hashed value. Moreover, typical Plasma designs use as  $H$  the so-called *Merkle-tree hashing* with  $tx_1, \dots, tx_m$  being the labels of the leaves (see the full version of this paper [11] for more on this technique). Thanks to this, every user can prove that his transaction was included in  $H(T)$  with a proof of length  $O(\log n)$ . If the operator does not include some  $tx_i$  in  $T$  then the user  $U$  that produced  $tx_i$  can always post  $tx_i$  directly to the blockchain. In this case using Plasma does not bring any benefits to  $U$  compared to just using the blockchain directly from the beginning. However, in reality it is expected that this is not going to happen often, especially since the operators are envisioned to be commercial entities that will charge some fee for their services.

What is much more problematic is the case when the malicious operator does *not* publish  $T$  off-chain. This situation is typically referred to as *data unavailability*. Note that data unavailability is subjective, i.e., the parties can have different views on whether it happened or not. This is because, unlike the situation on the blockchain, there may be no consensus on what was published off-chain. Moreover, there is no way to produce a proof that data unavailability happens: even if some parties complain on the blockchain that they did not receive  $T$  there is no way to determine (just by looking at the blockchain) whether they are right, or if they are just falsely accusing the operator. The situation when a disagreement between parties happens but there is no “blockchain-only” way to verify which party is corrupt, is commonly referred to as a *non-uniquely attributable fault*. Finally, some Plasma protocols require all the honest parties to immediately act on blockchain after a data unavailability attack happened. This is called “mass exit”, although (for the reasons explained in Sec. 1.1 in this paper we call it “large forced on-chain action”)

Plasma comes in many variants and has been discussed in countless articles (see Sec. 1.1). One of the most fundamental distinction is between the two types of Plasma systems: *Plasma Cash* and *Fungible Plasma*. As we explain in more detail in Sec. 1.1 they both serve for the “emulation” that we outlined above, but have different incomparable features. From one point of view Fungible Plasma is better than Plasma Cash since it is “fungible”, which means that the money can be arbitrarily divided and merged. On the other hand: Fungible Plasma suffers from some problems that Plasma Cash does not have, namely the adversary can cause a “non-uniquely attributable large forced mass actions” (we explain these notions Sec. 1.1). The cryptocurrency community has been unsuccessfully looking for a Plasma solution that

<sup>2</sup> In this paper we mostly use the name “Plasma” due to its brevity.

would have the benefits of both Plasma Cash and Fungible Plasma. The main result of this paper is that such Plasma cannot be achieved simultaneously. In other words, we show inherent limitations of the compression technique, at least for compressing blockchain financial transactions. Our paper can also be viewed as initiating the theoretical analysis of lower bounds for the smart contract protocols. We write more about our contribution in Sec. 1.2. First, however, let us provide some more introduction to blockchain and to Plasmas.

## 1.1 Introduction to Plasma

Let us start with providing some more background on blockchain and smart contracts. Blockchain can be viewed as a public *ledger* containing timestamped transactions that have to satisfy some correctness constraints. Moreover, several blockchains permit to execute the so-called *smart contracts* [27] (or simply: “contracts”), which informally speaking, are “self-executable” agreements described in form of computer programs. Examples of such blockchain platforms include *Ethereum*, *Hyperledger Fabric*, or *Cardano*. Typically, it is assumed that contracts are deterministic and have a public state. Moreover, they can own some coins. Executing a contract is done by posting transactions on the blockchain and it costs fees that depend on the computational complexity of the given operation, and on the amount of data that needs to be transmitted to the contract.

Let us now explain the basic idea of Plasma, and introduce some standard terminology. Since it is an informal presentation, we mix the definition of the protocol with its construction. In the formal sections of the paper these two parts are separated (the definition appears in Sec. 2, and the constructions in the full version of this paper [11]). As highlighted above Plasma address the scalability problem of blockchain by keeping the massive bulk of transactions outside of the blockchain (“off-chain”). The parties that are involved in the protocol rely on a smart contract that is deployed on the ledger of the underlying cryptocurrency, but they try to minimize interacting with it. Typically, this interaction happens only when the parties join and leave the protocol, or when they disagree. Since all parties know that in case of disagreement, disputes can always be settled on the ledger, there is no incentive for the users to disagree, and honest behavior is enforced.

In the optimistic case, when the parties involved in the protocol play honestly, and the off-chain transactions never hit the ledger, these protocols significantly reduce transaction fees and allow for instantaneous executions. Off-chain protocols also resemble an idea explored in cryptography around two decades ago under the name “optimistic protocols” [7, 1]. In this model the parties are given access to a trusted server that is “expensive to use”, and hence they do not want to contact it, unless it is absolutely necessary.

Plasma’s operator *Op* provides a “simulated ledger”, in which other parties can deposit their coins, and then perform operations between each other. The key requirement is that its users do not need to trust the operator, and in particular if they discover that she is cheating, then they can safely withdraw their funds. The latter is called an *exit* from the simulated ledger, and requires communication with the underlying ledger.

Plasma protocols come in different variants (see Sec. 1.1), however, they are all based on a single framework proposed in [26]. The parties that execute Plasma are: the *users*  $U_1, \dots, U_n$ , and the *operator* *Op*. Moreover, the parties have access to a contract on the blockchain. In our formal modeling this contract will be represented as a trusted interactive machine  $\Gamma$  with public state, owning some amount of coins. Each user  $U_i$  has some number of coins initially deposited in his Plasma account which is maintained by  $\Gamma$ . This number is called a *balance* and is denoted with  $b_i \in \mathbb{Z}_{\geq 0}$ . Users’ balances are changing dynamically during the execution of the protocol. The total number of coins owned by the contract  $\Gamma$  is

equal to the sum of all balances of its users. A vector  $\vec{b} := (b_1, \dots, b_n)$  is called a *Plasma chain*. When referring to the underlying blockchain (i.e. the one on which  $\Gamma$  is deployed) we use the term *main chain*. Note that the operator  $Op$  has no account and only facilitates transfers of the users. In some variants of Plasma (see Sec. 1.1) the operator blocks some amount of coins (called operator's *collateral*) that can be used to compensate the users their losses in case she misbehaves.

Let us briefly describe the different operations that parties of the Plasma protocol can execution during the lifetime of the system. We divide time into *epochs* (e.g. 1 epoch takes 1 hour). In the  $i$ th epoch the operator sends some information  $C_i$  to  $\Gamma$ . We can think of  $C_i$  as “compressed” information about the vector  $(b_1, \dots, b_n)$  containing the users' balances. By “compressed” we mean that  $|C_i|$  is much shorter than the description of  $(b_1, \dots, b_n)$ , and usually its length is constant in every epoch. We will refer to  $C_i$  as a “commitment” to  $(b_1, \dots, b_n)$ . The length of  $C_i$  is called the *commit size*.

In each epoch every user  $U_i$  can request to *exit*, by which we mean that all  $b_i$  coins from her Plasma account get converted to the “real” coins on the main chain, and she is no longer a part of this Plasma chain (which, in our formal modeling will be indicated by setting  $b_i := \perp$ ). Plasma's security properties guarantee that every user can exit with all the coins that she currently has in the given Plasma chain. It is often required that exiting can be done cheaply, and in particular that the total length of the messages sent by the exiting user to  $\Gamma$  is small. The amount of data that a user needs to send to  $\Gamma$  in order to exit the Plasma chain is called the *exit size*.

Finally, any two users of the same Plasma chain can make *transfers* between each other. Suppose  $U_k$  wants to transfer  $v$  coins from her account to  $U_j$ . This transfer operation involves only communicating with  $U_j$ , and with the operator  $Op$ , while no interaction with the contract  $\Gamma$  is needed. Under normal circumstances (i.e. when the operator is honest) the next Plasma block that is committed to the main chain will simply have  $v$  coins deduced from  $U_k$ 's account and  $v$  coins added to  $U_j$ 's account.<sup>3</sup>

### 1.1.1 Challenges in designing Plasma systems

The main challenge when designing a Plasma system is to guarantee that every user can exit with her money. This is usually achieved as follows: each  $C_i$  is a commitment to  $(b_1, \dots, b_n)$ , computed using a Merkle tree. An honest operator  $Op$  is obliged to obey the following rule:

**Explaining commitments** – each time  $Op$  sends  $C_i$  to  $\Gamma$ , she sends the corresponding  $\vec{b} := (b_1, \dots, b_n)$  to all the users.

Technically, sending  $\vec{b}$  to the users can be realized, e.g., by publishing it on the operator's web-page (i.e.: “off-chain”). Every user  $U_j$  can now check if she has the correct amount on her account and if  $C_i$  was computed correctly. Moreover, thanks to the properties of Merkle trees,  $U_j$  has a short proof of size  $O(\log(n))$  that  $b_j$  has been “committed” into  $C_i$ .

The above description assume that the operator is honest. If she is corrupt things get more complicated. Note that  $C_i$  sent by the operator to  $\Gamma$  is publicly known (due to the properties of the underlying blockchain). Hence, we can assume that all the users agree on whether  $C_i$  was published and what is its exact value. The situation is different when it comes to the vector  $\vec{b}$  that should be published off-chain. In particular, if  $\vec{b}$  has *not* been published, then the users have no way to prove this to  $\Gamma$ . This is because whether some data

<sup>3</sup> To keep things simple, in this paper we do not discuss things like “transfer receipts”, i.e., confirmations for the sender that the coins have been transferred.



has been published off-chain or not does not have a digital evidence that can be interpreted by  $\Gamma$ . This leads us to the following attack that can be carried out by a malicious operator, and is intensively studied by the cryptocurrency community (see, e.g., [3]).

**Data unavailability attack** – in this attack the corrupt operator publishes  $C_i$  but does not publish  $\vec{b}$ .

Note that this attack has no extra cost for the operator because from the point of view of the smart contract  $\Gamma$ , the operator behaves honestly, and hence the users cannot complain to  $\Gamma$  and request, e.g., that  $Op$  sends  $\vec{b}$  to  $\Gamma$ . Furthermore determining if this attack happened is “subjective”, i.e., every user  $U_j$  has to detect it herself. Moreover, in case of data unavailability it is impossible for  $\Gamma$  to determine whether  $U_j$  or  $Op$  is dishonest, since a sheer declaration of  $U_j$  that  $Op$  did not send him the data obviously cannot serve as a proof that it indeed happened. This leads to the following definition.

**Non-uniquely attributable faults** – this refers to the situation when the contract has to intervene in the execution of the protocol (because the protocol is under attack), but it unable to determine which party misbehaved (see, e.g., [2]).

Non-uniquely attributable faults appear typically in situations when a party claims that it has not received a message from another party. In contrast if a contract *is* able to determine which party is corrupt then we have a *uniquely*-attributable fault. A typical example of such a fault is when a party signs two contradictory messages. Unfortunately, non-uniquely attributable faults are hard to handle in real life, since it is not clear which party should pay the fees for executing the smart contract, or which party should be punished for misbehaving. In particular, what is unavoidable in such a case is that a malicious party  $P$  can force another participant  $P'$  to lose money on fees (potentially also loosing money herself). This phenomenon is known as *griefing* [2].

When a user realizes that the operator is dishonest, then she often needs to start quickly interacting with  $\Gamma$  in order to protect her coins. This action has to be done quickly, and has to be performed by each honest user. This leads to the following definition.

**Forced on-chain action of size  $\alpha$**  – this term refers to the situation that honest parties who did not intend to perform an exit are forced by the adversary to quickly interact with  $\Gamma$ , and the total length of the messages sent by them to  $\Gamma$  is  $\alpha$ . Informally, when  $\alpha$  is large (e.g.  $\alpha = \Omega(n)$ ) we say this is a *mass* forced on-chain action.

Note that this definition talks about all honest “parties”, and hence it includes also the case when the operator is honest, but it is forced to act because of the behavior of the corrupt users. Typically  $\alpha = \Omega(n)$  and by “quickly” we mean “1 epoch”. In most Plasma proposals [23, 26, 4] “interacting with the smart contract” means simply exiting the Plasma chain with all the coins. Hence, a more common term for this situation is “mass exit”. Since in our work we are dealing with the lower bounds, we need to be ready to cover also other, non-standard, ways of protecting honest users’ coins. For example, it could be the case that a user  $U_j$  does *not* exit immediately, but, instead, keeps her coins in a special account “within  $\Gamma$ ” and withdraws them much later. Of course, this requires interacting with  $\Gamma$  immediately, but, technically speaking does not require “exiting”. To capture such situations, we use the term “forced on-chain *action*”, instead of “mass *exit*”.

After a party announces an exit, we need to ensure that she is exiting with the right amount of coins. The main problem comes from the fact that we cannot require that users exit from the last  $C_i$  by sending the explanation for her balance  $b_i$  to  $\Gamma$ . This is because it could be the case that a given user does not know the explanation  $\vec{b}$  of  $C_i$  (due to the data unavailability attack). For a description of how this can be done in practice see [26, 4], or the full version of this paper [11].

Mass exits (or large forced on-chain actions) caused by data unavailability are considered a major problem for Plasma constructions. They are mentioned multiple times in the original Plasma paper [26] (together with some ad-hoc mechanism for mitigating them). They are also routinely discussed on “Ethereum’s Research Forum”<sup>4</sup>, with even conferences organized on this topic<sup>5</sup>. One of the main reasons why the mass exits are so problematic is that they may result in blockchain congestion (i.e., situations when too many users want to send transactions to the underlying blockchain). Moreover, the adversary can choose to attack Plasma precisely in the moments when the blockchain is already close to being congested (see, e.g., [30] for a description of real-life incident of the Ethereum blockchain congestion). She can also attack different Plasma chains (established over the same main chain) so their users simultaneously send large amounts of data to the blockchain. In order to be prepared for such events in real-life Plasma proposals it is sometimes suggested that the time  $T$  for reacting to data unavailability should be very large (e.g.  $T = 2$  weeks). This, unfortunately, has an important downside, namely that also an honest coin withdrawal requires time  $T$ .

Consider a non-fungible Plasma that supports coin identifiers from some set  $\mathcal{C}$ . In a non-fungible Plasma the Plasma chain is of a different form than before: instead of a vector of balances  $\vec{b}$ , it is a function  $f : \mathcal{C} \rightarrow \{U_1, \dots, U_n, \perp\}$  that assigns to every coin  $c \in \mathcal{C}$  its current owner  $f(c)$  (or  $\perp$  if the coin has been withdrawn). Similar to before the commitment to the value of  $f$  will be done using Merkle trees. Whenever a coin is withdrawn its identifier  $c$  is sent to the smart contract  $\Gamma$ , and hence it becomes public. This is important for the mechanism that prevents parties from stealing coins. To this end, each user  $U$  monitors  $\Gamma$ , and sends a complaint whenever some (corrupt) user  $U'$  tries to withdraw one of  $U$ ’s coins. For the contract  $\Gamma$  to decide if  $c$  belongs to  $U$  or  $U'$  can require some additional interaction, but the system is designed in such a way that the honest user is guaranteed that finally she will win such dispute. Hence, every malicious attempt to withdraw someone else’s coin will be stopped.

The main difference between Plasma Cash and Fungible Plasma is that in Plasma Cash every user has to “protect” only her own coins. Thanks to this, even in case of the data unavailability attack, each honest user  $U$  does *not* need to immediately take any action. Instead, she can just monitor  $\Gamma$ , and has to act only if someone tries to withdraw one of  $U$ ’s coins. Of course, the corrupt user can still force all the honest ones to quickly act on the blockchain. However, this requires much more effort from them than in Fungible Plasma, namely: they need to withdraw many coins of the honest users at once, hence forcing the honest users to react. This is “fairer” both honest and malicious users have to make similar effort. Most importantly, however, this attack has *uniquely-attributable faults*.

This advantage of Plasma Cash comes at a price, namely the “exit size” is not constant anymore, as it depends on the number of coins that a user has (since each coin has to be withdrawn “independently”). The Ethereum research community has been making some efforts to deal with this problem. One promising approach is to “compress” the information about withdrawn coins. For example one could assume that the identifiers in  $\mathcal{C}$  are natural numbers. Then a user  $U$  who owns coins from some interval  $[a, \dots, b]$  (with  $a, b \in \mathbb{N}$ ) could simply withdraw them by posting a message “User  $U$  withdraws all coins from the interval  $[a, \dots, b]$  (instead of withdrawing each  $i \in [a, \dots, b]$  independently). This, of course, works only if the coins that users own can be divided into such intervals. Some authors (in particular V. Buterin) have been suggesting “defragmentation” techniques for achieving

<sup>4</sup> Available at: [ethresear.ch](https://ethresear.ch).

<sup>5</sup> See: [ethresear.ch/t/data-unavailability-unconference-devcon4](https://ethresear.ch/t/data-unavailability-unconference-devcon4).

such a distribution of coins. This is based on the assumption that the parties periodically *cooperate* to “clean up” the system. Hence, it does not work in a fully malicious settings<sup>6</sup> (if the goal of the adversary is to prevent the cleaning procedure).

### 1.1.2 The landscape of Plasmas

Soon after the original groundbreaking work on Plasma [26], some concrete variants have been proposed. Some of them we already described in Sec. 1.1. Since this paper focuses on the *impossibility* results, we do not provide a complete overview of the many different variants that exist and what features they achieve. Plasma projects that are frequently mentioned in the media are Loom, Bankex, NOCUST and OmiseGO [28, 24]. This area is mostly developed by a very vibrant on-line community that typically communicates results in form of so-called “white-papers”, blog articles, or post on discussion forums (such as the “Ethereum Research Forum”, see footnote 4 on page 7). See also the diagram called “Plasma World Map” illustrating the different flavors of Plasma in the full version of this paper [11]. A notable exception are NOCUST and NOCUST-ZKP described in [21]. This work, up to our knowledge, is the first academic paper on this topic. It provides a formal protocol description (with several interesting innovations such as “Merkle interval trees”) and a security argument. Moreover, the authors of [21] describe a version of NOCUST that puts a collateral on the operator (this is done in order to achieve instant transaction confirmation). The authors of [21] (see also [16]) introduce the term “commit chains”. Yet, unfortunately, they do not define a full formal security model that we could re-use in our work.

Let us also mention some of the so-called “distributed exchanges” that look very similar to Plasma. One example is StarkDEX (informally described in [15]), which is also based on the idea of a central operator batching transactions using Merkle trees, and a procedure for the users to “escape” from the system if something goes wrong. This protocol uses non-interactive zero-knowledge protocols to ensure correctness of the operator’s actions (similar approach has been informally sketched in the original Plasma paper [26], and has been also used in NOCUST-ZKP [21]). While zero knowledge can be used to demonstrate that some data was computed correctly, it *cannot* be used to prove that the off-chain data *was published at all*. Consequently, the authors of this system also encountered the challenge of handling the data unavailability attack. Currently, in StarkDEX this problem is solved by introducing an external committee that certifies if data is available.<sup>7</sup> StarkDEX plans to eventually replace the committee-based solution with an approach that is only based on trusting the underlying blockchain. Our result however shows that in general this will be impossible, as long as fungibility and short exits are required (unless the operator puts a huge collateral).

## 1.2 Our contribution and organization of the paper

We initiate the study of lower bounds (or: “impossibility results”) in the area of off-chain protocols. Our results can also be viewed as a part of a general research program of “bringing order to Plasma”. We believe that the scientific cryptographic community can provide significant help in the efforts to systematize this area, and to determine the formal security guarantees of the protocols (in a way that is similar to the work on “Bitcoin backbone” [14], or more recently on “Mimblewimble” [13], state channels [9], or the Lightning Network [22]).

<sup>6</sup> See [ethresear.ch/t/plasma-cash-defragmentation](https://ethresear.ch/t/plasma-cash-defragmentation) and subsequent posts by Buterin on the Ethereum Research Forum.

<sup>7</sup> See their FAQs at <https://www.starkdex.io>.

Investigating the limits of what Plasma can achieve is part of this process. We focus on proving lower bounds that concern the necessity of mass forced on-chain actions, especially caused by the attack that have no uniquely attributable faults (as a result of data unavailability). This is motivated by the fact that such attacks are particularly important for the off-chain protocols: since the main goal of such protocols is to move the transactions *off*-chain, the necessity of quickly acting *on*-chain can be viewed as a big disadvantage.

We start with a formal definition of Plasma (this is done in Sec. 2). Since in this work we are interested only in the impossibility results, our definition is very restrictive for practical systems. By “restrictive” we mean that we make several assumptions about how the protocol operates. For example we have very strict synchronicity rules, and in particular we only allow the users to start the Plasma operations in certain moments (see “payment orders” and “exit orders” phases in Sec. 2.1). Obviously, such restrictions make our lower bounds only stronger, since they also apply to a more realistic model (without such restrictions). We believe that fully formalizing real-life Plasma (e.g. in the style of [12, 10, 22]) is an important future research project, but it is beyond the scope of this work.

Our main result is presented in Thm. 1, stated formally in Sec. 3. It states that in certain cases the adversary can force mass on-chain actions of the honest users of *any* Plasma system. One subtle point that we want to emphasize is that whenever we talk about “forcing actions” on the honest users, we mean a situation when the users that did *not* want to exit (in a given epoch) are forced to act on-chain. This is important, as otherwise our theorem would hold trivially (one can always imagine a scenario when lots of users decide to exit Plasma because of some other, external, reasons). The notion of “wanting to exit” is formalized by an *environment machine*  $\mathcal{Z}$  (borrowed from the *Universal Composability* framework [8]) that “orders” the parties to behave in certain way.

More formally, Thm. 1 states that in Plasma either there exists an attack that provokes a mass action, or there is an attack that requires a party that exits to post long messages on the blockchain (i.e. this Plasma has large exits). Moreover, both attacks have *no* uniquely attributable faults. Note that, strictly speaking, this theorem also covers Plasma systems where the commit size is large (even  $\Omega(n)$ )<sup>8</sup>, but in this case it holds trivially since the honest operator needs to send the large commitments to  $\Gamma$  even if everybody is honest (hence: there is an “unprovoked” mass action in every epoch).

The most interesting practical implication of this theorem is that it confirms the need for “two different” Plasma flavors, as long as the operator is not required to put aside a collateral of size comparable to the total amount of coins in the system<sup>9</sup>. One way to look at it is: either we want to have a Plasma system that does not have large exits, in which case we need to have (non-uniquely attributable) mass actions (this is Fungible Plasma/Plasma MVP); or alternatively we insist on having Plasma without such mass actions, but then we have to live with large exits (as in Plasma Cash). Our theorem implies that there is no Plasma that would combine the benefits of both Fungible Plasma and Plasma Cash, and hence can serve as a justification why both approaches are complementary. Before our result one could hope that the opposite is true and that, e.g., the only reason why Plasma Cash is popular is its relative simplicity (compared to Fungible Plasma). Besides of this reason, and the general scientific interest, we believe that our lower bound has some other important practical applications. In particular, lower bounds often serve as a guideline for constructing new systems or tweaking

<sup>8</sup> In practice, Plasma systems with unbounded commit size are not interesting since they do not bring any advantages to the users. Moreover, they can be trivially constructed just by putting every transaction on the main chain.

<sup>9</sup> This is clearly impractical for most of the applications. Actually most of Plasma constructions assume no such collateral at all.

the definitions. We hope it will contribute to consolidate the countless research efforts in constructing new Plasma systems<sup>10</sup> and simplify identifying proposals that are not sound (e.g., because they claim to achieve the best of both worlds).

Let us also stress that our Thm. 1 does *not* rely on any assumptions of complexity-theoretic type and does *not* use a concept of “black-box separations” [18]. This means that the lower bound that we prove cannot be circumvented by introducing any kind of strong cryptographic assumptions. Hence, of course, it also holds for Plasmas that use non-interactive zero-knowledge (like NOCUST-ZKP [21] or StarkDEX [15]). Moreover, we manage to generalize our lower bound. Thm. 1 even holds for Plasma systems where the operator deposits a certain amount of coins for compensating parties for malicious behavior (e.g., it could be used when a malicious operator does not explain commitments).

For completeness, we also describe (in the full version of this paper [11]) “positive” results, i.e., two protocols that satisfy our security definition (Plasma Cash and Fungible Plasma). We stress that we do not consider it to be a part of our main contribution, and we do not claim novelty with these constructions, as they strongly rely on ideas published earlier (in particular [26, 5, 23, 4, 21, 15]).

**Notation.** For a formal definition of an *interactive (Turing) machine* and a *protocol*, see, e.g., [8]. In our modeling the communication between the parties is synchronous and happens in *rounds* (see Section 2). During the execution of the protocol a party  $P$  may send messages to a party  $P'$ . A *transcript* of the messages sent from  $P$  to  $P'$  is a sequence  $\{(m_i, t_i)\}_{i=1}^\ell$ , where each  $m_i$  was sent by  $P$  to  $P'$  in the  $t_i$ -th round. A transcript of messages sent from some set of parties to a different set of parties is a sequence  $\{(W_i, W'_i, m_i, t_i)\}_{i=1}^\ell$ , where each  $m_i$  was sent by  $W_i$  to  $W'_i$  in the  $t_i$ -th round. By the *length of a transcript* we mean its bit-length (in some fixed encoding). We sometimes refer to it also as *communication size* (between the parties).

## 2 Plasma Payment Systems

A *Plasma payment system* (or “Plasma” for short) is a protocol  $\Pi$  consisting of a randomized non-interactive machine  $\Psi$  representing the *setup* of the system; deterministic<sup>11</sup> interactive poly-time machines  $U_1, \dots, U_n, Op$  representing the *users* and the *operator* of the Plasma system (respectively); and a deterministic interactive poly-time machine  $\Gamma$ , which represents the *Plasma contract*. We use the notation  $\mathcal{U} = \{U_1, \dots, U_n\}$  to refer to the set of users of the system. The contract machine  $\Gamma$  has no secret state, and moreover its entire execution history is known to all the parties. We can think of it as a Turing machine that keeps the entire log of its execution history, and moreover all the other parties in the system have a (read-only) access to this log. The Plasma system comes with a parameter  $\gamma \in \mathbb{R}_{\geq 0}$  called *operator’s collateral fraction*. Informally, this parameter describes the amount of coins that are held by the operator as a “collateral” (as a fraction of user’s coins). These coins can be used to cover users’ losses if the operator misbehaves. This is formally captured in Section 2.2 (see “limited responsibility of the operator”). If  $\gamma = 0$  then we say that the operator is *not collateralized*. We introduce the notion of collateral in order to make our results stronger and to cover also cases of real-life systems that have such a collateral (e.g., NOCUST, see Section 1.1).

<sup>10</sup> see <https://ethresear.ch/c/plasma/>.

<sup>11</sup> We assume that these machines are deterministic, since all their internal randomness will be passed to them by  $\Psi$ .

The protocol is attacked by a randomized poly-time adversary  $\mathcal{A}$ . We assume that  $\mathcal{A}$  can corrupt any number of users and the operator except the contract  $\Gamma$  (hence  $\Gamma$  can be seen as a trusted third party). Once  $\mathcal{A}$  corrupts a party  $P$ , she learns all its secrets, and takes full control over it (i.e. she can send messages on behalf of  $P$ ). A party that has not been corrupted is called *honest*. An execution of a Plasma payment system  $\Pi$  is parametrized by the *security parameter*  $1^\lambda$ .

To model the fact that users perform actions, we use the concept of an environment  $\mathcal{Z}$  (which is also a poly-time machine) that is responsible for “orchestrating” the execution of the protocol. The environment can send and receive messages from all the parties (it also has full access to the state of  $\Gamma$ ). It also knows which parties are corrupt and which are honest. For an adversary  $\mathcal{A}$  and an environment  $\mathcal{Z}$ , a pair  $(\mathcal{A}, \mathcal{Z})$  will be called an *attack* (on a given Plasma system  $\Pi$ ). The contract machine  $\Gamma$  can output special messages (*attribute-fault*,  $P$ ) (where  $P \in \mathcal{U} \cup \{Op\}$ ). In this case we say that  $\Gamma$  *attributed a fault to*  $P$ . We require that the probability that  $\Gamma$  attributes a fault to an honest party is negligible in  $\lambda$ . An attack  $(\mathcal{A}, \mathcal{Z})$  *has no attributable faults* if the probability that  $\Gamma$  attributes a fault to some party is negligible.

## 2.1 Protocol operation

Let us now describe the general scheme in which a Plasma payment protocol operates. In this section, we focus only on describing what messages are sent between the parties. The “semantics” of these messages, and the security properties of the protocol are described in Section 2.2. We assume that all the parties are connected by authenticated and secret communication channels, and a message sent by a party  $P$  in the  $i$ th round, arrives to  $P'$  at the beginning of the  $(i + 1)$ th round. The communication is synchronous and happens in *rounds*. It consists of three stages, namely: “setup”, “initialization”, and “payments”. The execution starts with the *setup stage*. In this stage parameter  $1^\lambda$  is passed to all the machines in  $\Pi$ . Upon receiving this parameter, machine  $\Psi$  samples a tuple  $(\psi_{U_1}, \dots, \psi_{U_n}, \psi_{Op}, \psi_\Gamma)$  (where each  $\psi_P \in \{0, 1\}^*$ ). Then for each  $P \in \{U_1, \dots, U_n, Op, \Gamma\}$  the string  $\psi_P$  is passed to  $P$ . Afterwards, the parties proceed to the *initialization stage*. In this stage the environment generates a sequence  $(a_1^{\text{init}}, \dots, a_n^{\text{init}})$  of non-negative integers and passes it to the contract  $\Gamma$  (recall that the state of  $\Gamma$  is public, and hence, as a consequence, all the parties in the system also learn the  $a_i^{\text{init}}$ ). Then the protocol proceeds to the *payment stage*. This stage consists of an unbounded number of *epochs*. Each  $i$ th epoch (for  $i = 1, 2, \dots$ ) is divided into two *phases*.

**Payment phase.** In this phase the environment sends a number of *payment orders* to the users (for simplicity we assume that this happens simultaneously in a single round). Each order has a form of a message “(send,  $v, U_i$ )”, where  $v \in \mathbb{Z}_{\geq 0}$ , and  $U_i \in \mathcal{U}$ . It can happen that some users receive no payment orders in a given epoch. It is also ok if a user receives more than one order in an epoch. Informally, the meaning of these messages is as follows: if a user  $U_j$  receives a “(send,  $v, U_i$ )” message, then she is ordered to transfer  $v$  coins to user  $U_i$ . We require that this message can only be sent if none of  $b_i$  and  $b_j$  are equal to  $\perp$  (i.e.: if none of  $U_i$  and  $U_j$  “exited”, see below). The parties execute a multiparty sub-protocol. During this executions some of the users send a message “(received,  $v, U_i$ )” to the environment  $\mathcal{Z}$ . This sub-protocol ends when  $\Gamma$  outputs a message *payments-processed*.

**Exit phase.** In this phase the environment sends *exit orders* to some of the users (again: this happens in a single round). Each such order is simply a message “exit”. Informally, sending this message to some  $U_i$  means that  $U_i$  is ordered to exit the system with all



her coins. The environment can send an exit message to  $U_i$  only if  $b_i \neq \perp$  (i.e.  $U_i$  has not already “exited”, see below). The parties again execute a multiparty protocol. The protocol ends when  $\Gamma$  outputs a sequence

$$\{(\text{exited}, U_{i_j}, v_{i_j})\}_{j=1}^m, \quad (1)$$

where  $m$  is some non-negative integer, and each  $U_{i_j} \in \mathcal{U}$  and  $v_{i_j} \in \mathbb{Z}_{\geq 0}$ . For each  $U_{i_j}$  in Eq. (1) we say that  $U_{i_j}$  *exited (with  $v_{i_j}$  coins)*, and we let  $b_{i_j} := \perp$ . We require that no party can exit more than once. In other words: it cannot happen that two messages  $(\text{exited}, U_i, v)$  and  $(\text{exited}, U_i, v')$  are issued by  $\Gamma$ .

We make some assumptions on the communication between the parties. Informally we require that if  $U$  and  $U'$  are some honest users, then the procedure of transferring coins from  $U$  to  $U'$  is done by a “sub-protocol” involving only parties in the set  $U$  and  $U'$ . Since we do not have a concept of “sub-protocol” this is formalized as follows:

**Communication locality.** Two honest users  $U$  and  $U'$  exchange messages only in epochs in which they do transactions between each other (i.e. a message  $(\text{send}, U, v)$  is sent by  $\mathcal{Z}$  to  $U'$ , for some  $v$ ).

This requirement is very natural since Plasma is supposed to work even when an arbitrary set of users is corrupt. Hence, relying on the other users’ help in financial transfers would be impractical. Up to our knowledge all “pure” Plasma proposals in the literature satisfy this requirement. On the other hand: it may *not* hold if we incorporate some techniques that assume some type of cooperation between larger sets of parties (e.g. consensus mechanisms). Examples include: Buterin’s Plasma Chash defragmentation (where a large set of users has to regularly cooperate in order to “clean-up” the system), and StarkDEX’s “data availability committee” (see Section 1.1), if we treat the committee members as “users”. One way to view our result is that it implies that such techniques are inherent for every fungible Plasma.

## 2.2 Security properties

During the interaction with the protocol, the environment keeps track of balances of *honest* users (we do not define balances of dishonest users). Formally, for each honest user  $U_i$  it maintains a variable  $b_i \in \mathbb{Z}_{\geq 0} \cup \{\perp\}$  (a *balance* of  $U_i$ ), where the symbol “ $\perp$ ” means that a party exited. It also maintains a variable  $t \in \mathbb{Z}_{\geq 0}$  (initially set to 0) that is used to keep track on the amount of coins that have been withdrawn. The rules for maintaining these variables are as follows. Initially, for each  $i := 1, \dots, n$  the environment  $\mathcal{Z}$  lets  $b_i := a_i^{\text{init}}$ . Whenever  $\Gamma$  outputs  $(\text{exited}, U_i, v)$  (for some  $U_i$  and  $v$ ) we let  $b_i := \perp$  and increment  $t$  by  $v$ . Each time  $\mathcal{Z}$  receives a message  $(\text{received}, v, U_i)$  from some *honest*  $U_j$ , it adds  $v$  to  $b_j$  (recipient balance) and, if  $U_i$  (sender) is honest too, subtracts  $v$  from  $b_i$ . We require that the environment never issues an order if  $U_i$  or  $U_j$  exited (i.e. if  $b_i = \perp$  or  $b_j = \perp$ ). The environment also never sends an order exit to the same user more than once, and it never sends exit order to a user  $U_i$  that already exited (i.e. such that  $b_i = \perp$ ). We have the following security properties.

**Responsiveness to “send” orders.** Suppose  $Op, U_i$ , and  $U_j$  are honest, and the environment issued an order  $(\text{send}, v, U_i)$  to  $U_j$  then in the same epoch party  $U_i$  sends a message  $(\text{received}, v, U_j)$  to the environment.

**Correctness of “received” messages.** Suppose  $U_i$  and  $U_j$  are honest and  $U_i$  outputs a message  $(\text{received}, v, U_j)$ , then environment has issued an order  $(\text{send}, v, U_i)$  to  $U_j$  in the same epoch.



**Responsiveness to “exit” orders.** Suppose  $U_i$  is honest and the environment issued an order exit to  $U_i$  then in the same epoch  $\Gamma$  outputs a message  $(\text{exited}, U_i, v)$  (for some  $v$ ).

**No forced exits if operator honest.** Suppose  $Op$  and  $U_i$  are honest and  $\Gamma$  outputs message  $(\text{exited}, U_i, v)$  at epoch  $r$ , then environment has sent the order exit to  $U_i$  in the same epoch.

**Fairness for the users.** If  $\Gamma$  outputs a message  $(\text{exited}, U_i, v)$  (for some honest  $U_i$ ) then  $v \geq b_i$  (where  $b_i$  is the current balance of  $U_i$ ).

**Limited responsibility of the operator.** If the operator is honest, then the total amount of coins that are withdrawn from the system is at most  $a_1^{\text{init}} + \dots + a_n^{\text{init}}$ . Otherwise (if she is dishonest) the total amount of coins that are withdrawn from the system is at most  $\lceil (1 + \gamma)(a_1^{\text{init}} + \dots + a_n^{\text{init}}) \rceil$ . This definition captures the notion of operator’s collateral, and the fact that it is used (to cover users’ losses) if the operator is caught cheating.

If an attack  $(\mathcal{A}, \mathcal{Z})$  succeeds to violate any of the requirements from this section, then we say that  $(\mathcal{A}, \mathcal{Z})$  broke a given Plasma payment system. We say that  $\Pi$  is secure if for every environment  $(\mathcal{A}, \mathcal{Z})$  the probability that  $\mathcal{A}$  breaks  $\Pi$  is negligible in  $1^\lambda$ .

As explained in the introduction, certain attacks on Plasma are of particular importance, due to the fact that they are hard to handle in real life. We say that  $(\mathcal{A}, \mathcal{Z})$  force an on-chain action of size  $M$  (in some epoch  $i$ ) if the following happened. Let  $T$  be the set of honest parties that did not receive any order from  $\mathcal{Z}$  in epoch  $i$ . Then the total length of messages sent by parties from  $T$  to  $\Gamma$  is at least  $M$ . As explained in the introduction, the term that is more standard than “forced on-chain action” is “mass exit”. See 1.1 for a discussion why “forced on-chain action” is a better term when impossibility results are considered.

### 3 Our main result

We now present Thm. 1, which is the main result of this paper. The main implication of this theorem is that for every non-collateralized Plasma system there exists an attack that provokes a mass forced on-chain action, i.e., it forces the honest users to make large communication with the contract even if they did not receive any exit order from the environment (see point 1 in the statement of the theorem), unless a given Plasma system has large exits (point 2). Moreover, this can be done by an attack that has no uniquely attributable faults. This fact cannot be circumvented by putting a collateral on the operator, unless this collateral is very large.

► **Theorem 1** (Mass forced on-chain actions or large exits without uniquely attributable faults are necessary). *Let  $\Pi$  be a secure Plasma payment system with  $n$  users and let  $\gamma \geq 0$  be the operator’s collateral fraction. Then either*

1. *there exists an attack on  $\Pi$  that causes a forced on-chain action of size greater than  $(n - \lceil \gamma n \rceil \cdot \log_2 n - 5)/4$  with probability at least  $1/16 + \text{negl}(\lambda)$ , or*
2. *there exists an attack on  $\Pi$  such that one honest user, when ordered to exit by the environment, makes communication to  $\Gamma$  of size at least  $(n - \lceil \gamma n \rceil \cdot \log_2 n - 5)/4$  with probability at least  $1/16 + \text{negl}(\lambda)$ .*

*Moreover, both attacks have no uniquely attributable faults.*

One way to look at this theorem is as follows. First, consider a non-collateralized Plasma, i.e., assume that  $\gamma = 0$ . Let  $\mathcal{P}^1$  be a class of non-collateralized Plasma that with overwhelming probability do not have uniquely attributable forced on-chain actions (of any size larger than 0). In this case point 1 cannot hold, and hence, every Plasma  $\Pi \in \mathcal{P}^1$  needs to satisfy point 2. This means that there exists an attack on every  $\Pi \in \mathcal{P}^1$  such that one honest user, when

ordered to exit by the environment, makes communication to  $\Gamma$  of size at least  $(n - 5)/4$  with probability around  $1/16$ . Or, in other words: every Plasma from class  $\mathcal{P}^1$  must have a large exist size with noticeable probability. We know Plasma with such properties: it is essentially Plasma Cash (see the full version of this paper [11])

On the other hand, let  $\mathcal{P}^2$  be a class of non-collateralized Plasmas that with high probability have no large exits, in the sense of point 2 of Thm. 1. This means that point 1 has to hold, which implies that every  $\Pi \in \mathcal{P}^2$  needs to have large (at least around  $(n - 5)/4$ ) non-uniquely attributable mass forced on-chain actions. Plasma with such properties is called Fungible Plasma (see the full version of this paper [11]) Hence, informally speaking, Thm. 1 states that we cannot have the “best of two types of Plasma” simultaneously.

If we consider non-zero collaterals, i.e., we let  $\gamma > 0$  then the situation does not improve much, unless the collateral fraction is large, i.e., the total collateral blocked by the operator is at least around  $n \cdot \gamma = n/\log_2 n$ <sup>12</sup>. This essentially means that we cannot get around the bounds from Thm. 1 by introducing collateral, unless the amount of coins blocked in operator’s collateral is of roughly the same order as the total amount of coins stored by the users.

Note that even trivial versions of Plasma “fit” into Thm. 1. For example, consider Plasma in which the operator always puts all the transactions on-chain. Of course, the details would need to be worked out, but clearly such a Plasma can be made secure. The existence of such a trivial Plasma does not contradict our Thm. 1, since it clearly satisfies point 1: a large number of transfers in one epoch will cause a forced mass on-chain action (by the operator). The same also holds if every user needs to put each transaction on-chain.

## 4 Proof of Theorem 1

Before we present the proof let us introduce some auxiliary machinery. This is done in the next section.

### 4.1 Isolation scenario

Let  $\Pi$  be a Plasma payment system, let  $\mathcal{Z}$  be an environment, and let  $\mathcal{W}$  be some subset of the users of  $\Pi$ . We now introduce a procedure that we call *isolation of  $\mathcal{W}$* . In this scenario  $\Pi$  is executed as in the normal execution, except that we “isolate” the users  $\mathcal{W} \subseteq \mathcal{U}$  from the operator. More precisely: all the messages sent between any  $U \in \mathcal{W}$  and the operator  $Op$  are dropped, i.e., they never arrive to the destination. This scenario can be viewed as an “attack” although it does not fit into the framework from Section 2.1, since it violates the assumption that messages sent by an honest party to another honest party always arrive to the destination.

Although the isolation scenario cannot be performed within our model, it can be “emulated” by corrupting either the operator  $Op$ , or the users from  $\mathcal{W}$ . In the first case we corrupt the operator and instruct him to behave as if she was honest, except that she does not send messages to the users in  $\mathcal{W}$  and ignores all messages sent by these users. This will be called the *data unavailability (DU) attack against  $\mathcal{W}$  by the operator*. In the second, symmetric case (the *pretended data unavailability (PDU) attack by  $\mathcal{W}$  on the operator*) we corrupt the users in  $\mathcal{W}$ . Then, every user  $U \in \mathcal{W}$  behaves as if she was honest, except that she does not send messages to  $Op$ , and ignores all messages from  $Op$ .

<sup>12</sup> This is because we need to have  $\gamma \approx 1/\log_2 n$  to make the expression “ $(n - \lceil \gamma n \rceil \cdot \log_2 n - 5)/4$ ” equal to 0.

If this is the only type of malicious behavior, then “from the point of view” of all the other parties, and, most importantly, from the point of view of the contract machine  $\Gamma$ , it is impossible to say who is corrupt (the users in  $\mathcal{W}$  or the operator  $Op$ ). More precisely, we have the following.

► **Observation 1.** *Let  $\Pi$  be a Plasma payment system and consider the attack that isolates users in some set  $\mathcal{W}$  from the operator. Let  $\mathcal{Z}$  be an arbitrary environment and let  $\mathcal{T}_{\text{isolate}}^{\mathcal{W}, \mathcal{Z}}$  be the random variable denoting the transcript of messages received by  $\Gamma$ . Moreover, let  $\mathcal{T}_{\text{PDU}}^{\mathcal{W}, \mathcal{Z}}$  and  $\mathcal{T}_{\text{DU}}^{\mathcal{W}, \mathcal{Z}}$  be the random variable denoting the transcripts of messages received by  $\Gamma$  in the PDU attack and in the DU attack (respectively), both with environment  $\mathcal{Z}$ . Then  $\mathcal{T}_{\text{DU}}^{\mathcal{W}, \mathcal{Z}} \stackrel{d}{=} \mathcal{T}_{\text{isolate}}^{\mathcal{W}, \mathcal{Z}} \stackrel{d}{=} \mathcal{T}_{\text{PDU}}^{\mathcal{W}, \mathcal{Z}}$ .*

This fact is useful in the proof of the following simple lemma.

► **Lemma 1.** *Fix an arbitrary Plasma  $\Pi$ . Let  $\mathcal{W}$  be some set of users. Suppose  $\mathcal{A}$  performs a DU attack against  $\mathcal{W}$  or a PDU attack by  $\mathcal{W}$  (either by corrupting the operator or by corrupting the users), and let  $\mathcal{Z}$  be arbitrary. Then the attack  $(\mathcal{A}, \mathcal{Z})$  has no uniquely attributable faults.*

**Proof.** From the security of  $\Pi$  we get that if the users are corrupt then the probability that  $\Gamma$  attributes a fault to them is negligible. Symmetrically, if the operator is corrupt then the probability that  $\Gamma$  attributes a fault to her is negligible. By Observation 1 the transcripts of messages received by  $\Gamma$  in both attacks are distributed identically, so the probability that  $\Gamma$  attributes *any* fault has to be negligible. ◀

## 4.2 Proof overview

Fix some secure Plasma payment system  $\Pi$  that works for  $n$  users. We construct either an attack such that

$$\Pr \left[ \begin{array}{l} \text{the set of all honest users makes communication to } \Gamma \text{ of size} \\ \text{at least } (n - \lceil \gamma n \rceil \cdot \log_2 n - 5)/4 \\ \text{(without receiving an exit order from the environment)} \end{array} \right] \geq 1/16 + \text{negl}(\lambda), \quad (2)$$

or an attack such that

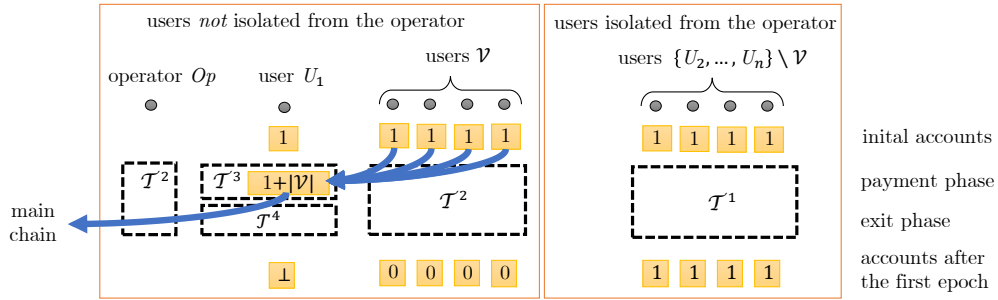
$$\Pr \left[ \begin{array}{l} \text{user } U_1, \text{ when ordered to exit by the environment, makes} \\ \text{communication to } \Gamma \text{ of size at least} \\ (n - \lceil \gamma n \rceil \cdot \log_2 n - 5)/4 \end{array} \right] \geq 1/16 + \text{negl}(\lambda). \quad (3)$$

In both of these attacks the amount of coins given to the users is  $n$ , but our proof can be generalized to cover cases when it is required that the amount of coins is larger than  $n$  (we comment more on this at the end of Section 4). On the other hand, the proof does not go through in the (unrealistic) case when this amount is very small (sublinear in  $n$ ).

The attacks that we construct in both cases ((2) and (3)) have no uniquely attributable faults. Note that for  $n \leq 5$  Eq. (3) holds trivially, and therefore we can assume that  $n > 5$ . Let  $\Upsilon$  denote the family of all *non-empty proper* subsets of  $\{U_2, \dots, U_n\}$ , i.e. sets  $\mathcal{V}$  such that  $\emptyset \subsetneq \mathcal{V} \subsetneq \{U_2, \dots, U_n\}$  (note that  $U_1 \notin \mathcal{V}$ ). Since we assumed that  $n > 5$  we have that  $\log |\Upsilon| = \log_2(2^{n-1} - 2) \geq n - 2$ , and, in particular,  $\Upsilon$  is non-empty. In the proof we construct an experiment (denoted by  $\text{Exp}(\mathcal{V})$  and presented in details in the full version of this paper [11]) and analyze its performance, assuming that  $\mathcal{V}$  is sampled uniformly at random from  $\Upsilon$ . Depending on this analysis, the experiment  $\text{Exp}(\mathcal{V})$  can be “transformed” into an attack that satisfies Eq. (2) or Eq. (3).

Experiment  $\text{Exp}(\mathcal{V})$  “simulates” an execution of two epochs of Plasma II. In the first epoch the adversary isolates the users in  $\{U_2, \dots, U_n\} \setminus \mathcal{V}$  from the operator (in the attacks that we construct later this will be done either by corrupting these users, or the operator). The environment gives 1 coin to each user  $U \in \mathcal{U}$ . Then, in the “payment” phase of the first epoch all the users from  $\mathcal{V}$  transfer their coins to  $U_1$ . In the “exit” phase of the first epoch user  $U_1$  receives an exit order from the environment and consequently exits with all her coins. Note that in the first epoch every party behaved honestly (except of the isolation attack against the users in  $\{U_2, \dots, U_n\} \setminus \mathcal{V}$ ), and hence  $U_1$  is guaranteed to successfully exit with her coins (she has 1 such coin from the “initialization” phase, and  $|\mathcal{V}|$  coins that were transferred to her by the users in  $\mathcal{V}$ ).

Of course the honest parties from  $\{U_2, \dots, U_n\} \setminus \mathcal{V}$  will usually realize that they are isolated from the operator. As a reaction to this they may send some messages to  $\Gamma$ . This, in turn can provoke the other parties to react by sending their messages to  $\Gamma$ . Hence, in general there can be a longer interaction between all the parties and  $\Gamma$  in this phase. Let  $\mathcal{T}^1$  be the transcript of the messages sent by the users in  $\{U_2, \dots, U_n\} \setminus \mathcal{V}$  to  $\Gamma$  in both phases, let  $\mathcal{T}^2$  be the messages sent by the users in  $\mathcal{V}$  and the operator to  $\Gamma$  in both phases, let  $\mathcal{T}^3$  be the messages sent by  $U_1$  to  $\Gamma$  in the “payment” phase, and finally let  $\mathcal{T}^4$  be the messages sent by  $U_1$  to  $\Gamma$  in the “exit” phase. The first epoch of the experiment  $\text{Exp}(\mathcal{V})$  and the transcripts are depicted on Fig. 1.



■ **Figure 1** The first epoch of the experiment  $\text{Exp}(\mathcal{V})$ . Gray circles denote the parties, and the  $\mathcal{T}^i$ 's denote the transcripts of the communication with  $\Gamma$  (see, e.g., Section 4.2 for their definitions).

Before discussing the second epoch of the experiment, let us note that in the first epoch the only way in which we deviate from the totally honest execution is the “isolation” of  $\{U_2, \dots, U_n\} \setminus \mathcal{V}$ . This will later allow us to be “flexible” and corrupt different sets of parties ( $\{Op\}$  or  $\{U_2, \dots, U_n\} \setminus \mathcal{V}$ ) depending on the results of our analysis of  $\text{Exp}(\mathcal{V})$ . This will be different in the second epoch, where we always assume that parties from  $\mathcal{V}$  are corrupt. This is ok because while constructing the attacks that satisfy (2) or (3) we will only use the first epoch of  $\text{Exp}(\mathcal{V})$ . The only reason to have the second epoch of  $\text{Exp}(\mathcal{V})$  is to make sure that the users have to send large amounts of data to  $\Gamma$  during the first epoch, as otherwise corrupt  $\mathcal{V}$  can steal the money (in the second epoch).

Let us now present some more details of the second epoch. Initially we corrupt all the users from  $\mathcal{V}$  and “rewind” them to the state that they had at the beginning of the first epoch. This is done in order to let them “pretend” that they still have their coins. We then let all of them try to (“illegally”) exit with these coins. Technically, “rewinding a user  $U$ ” is done via a procedure denoted  $\text{Reconstruct}_U$ . This procedure outputs the state that  $U$  would have at the end of the “payment” phase if she did not transfer her coins to  $U_1$ . To make it look consistent with the state of  $\Gamma$ , this procedure takes as input the transcripts defined

above. Then, each user  $U \in \mathcal{V}$  tries to exit (in the “exit” phase) from her state computed by  $\text{Reconstruct}_U$ . Also the honest users try to exit (they receive an “exit” order from the environment). Let  $\mathcal{Q}$  be the set of users that managed to exit with at least 1 coin. From the security of Plasma we get that  $\mathcal{Q}$  is equal to the set of honest users ( $\{U_2, \dots, U_n\} \setminus \mathcal{V}$ ) plus a small (of size at most  $\lceil \gamma n \rceil$ ) subset  $\mathcal{D}$  of dishonest users.

The key observation is now that all that is needed to “simulate” the second epoch of  $\text{Exp}(\mathcal{V})$  are the transcripts  $\mathcal{T}^1, \mathcal{T}^2, \mathcal{T}^3$ , and  $\mathcal{T}^4$ . On the other hand  $\mathcal{V}$  can be approximately computed from  $\mathcal{Q}$  (i.e., we can compute  $\mathcal{V}$  with elements  $\mathcal{D}$  missing, where  $|\mathcal{D}| = \lceil \gamma n \rceil$ ). Hence the variable  $(\mathcal{T}^1, \mathcal{T}^2, \mathcal{T}^3, \mathcal{T}^4)$  carries enough information to “approximately” describe  $\mathcal{V}$ . Thanks to this we can construct a “compression” algorithm that “compresses” a random  $\mathcal{V} \leftarrow \mathfrak{S} \Upsilon$  by simulating the first epoch of  $\text{Exp}(\mathcal{V})$  and obtaining  $(\mathcal{T}^1, \mathcal{T}^2, \mathcal{T}^3, \mathcal{T}^4)$  and then “decompresses” it by simulating the second epoch, and computing the output as  $\mathcal{V} := \{U_2, \dots, U_n\} \setminus \mathcal{Q}$  (the additional  $\lceil \gamma n \rceil$  elements can be simply listed as an additional output of  $\mathsf{C}$  and passed to  $\mathsf{D}$  as input that has to be added to the output of  $\mathsf{D}$ ).

On the other hand, clearly (for completeness we show this fact in the full version of this paper [11]), a random  $\mathcal{V} \leftarrow \mathfrak{S} \Upsilon$  with high probability cannot be compressed to a string that is significantly shorter than  $\log |\Upsilon| \geq n - 2$ . This implies that with a noticeable probability  $|(\mathcal{T}^1, \mathcal{T}^2, \mathcal{T}^3, \mathcal{T}^4)| \approx n - \lceil \gamma n \rceil \log_2 n$ , where  $\lceil \gamma n \rceil \log_2 n$  is the number of bits needed to describe set  $\mathcal{D}$ .

Obviously, the above fact implies that for at least one  $i \in \{1, \dots, 4\}$  we have that  $\mathcal{T}^i \geq (n - \lceil \gamma n \rceil \log_2 n)/4$  with noticeable probability for concrete parameters). The rest of the proof of Thm. 1 is based on the case analysis of the implications of “ $\mathcal{T}^i \geq n/4$ ” for different  $i$ ’s. More concretely, we show that in the first three cases ( $i = 1, 2$ , and  $3$ ) we can construct attacks that satisfy Eq. (2), and in case  $i = 4$  – an attack that satisfies Eq. (3). All these attacks are based on the experiment  $\text{Exp}(\mathcal{V})$ , but are only using its first epoch. In the proof we exploit the fact that the only malicious behavior that happens in this epoch is the “isolation” (i.e., not sending messages). Hence, we can use Observation 1 and “switch” between scenarios when different groups of parties are corrupt (while still getting the same transcripts  $\mathcal{T}^i$ ). Moreover these attacks do not have uniquely attributable faults.

The detailed proof of Thm. 1 can be found in the full version of this paper [11].

► **Remark 1.** Our proof would also go through even if the total balance of the users  $a$  is arbitrarily large. The only difference would be that instead of giving 1 coin to every user, the environment would give to each user  $U_i$  (for  $i > 1$ )  $\lfloor a/n \rfloor$  coins, and to user  $U_1$  the environment would give the remaining coins (say). The rest of the proof would be essentially identical to the proof of Thm. 1.

► **Remark 2.** Although the attack presented requires two epochs, the second epoch only captures the scenario where the underlying protocol is insecure and hence it can be seen as a “thought experiment”. In other words, if the honest parties do not make large communication with the contact  $\Gamma$  in the first epoch, they risk losing their coins in the second epoch. Therefore, under the assumption that the plasma system is indeed secure and consequently parties make large communication with  $\Gamma$  in the first epoch, the adversary cannot steal any coins in the second epoch and hence the second epoch would become obsolete.

## Conclusion

The main contribution of this work is that we have shown that the distinction between Plasma Cash and Fungible Plasma is inherent, i.e., we ruled out the possibility of constructing Plasma that combines benefits of both Plasmas. We believe that, besides of the general

scientific interest, our work (especially ruling out existence of some Plasma constructions) can help the practical blockchain community in developing Plasma protocols, and in general can bring more understanding in what is possible and what is impossible in the area of off-chain protocols, and under what assumptions. It can also serve as a formal justification why “hybrid” approaches (such as “rollups”) [6] may be needed in real life. We also hope that this work may expand the scope of theory by identifying a new area where theoretical lower bounds can have direct impact on the real life problems.

---

## References

- 1 N. Asokan, Matthias Schunter, and Michael Waidner. Optimistic protocols for fair exchange. In Richard Graveman, Philippe A. Janson, Clifford Neuman, and Li Gong, editors, *CCS '97, Proceedings of the 4th ACM Conference on Computer and Communications Security, Zurich, Switzerland, April 1-4, 1997*, pages 7–17. ACM, 1997. doi:10.1145/266420.266426.
- 2 Vitalik Buterin. A note on limits on incentive compatibility and griefing factors. URL: [https://vitalik.ca/files/extortion\\_griefing\\_bounds.pdf](https://vitalik.ca/files/extortion_griefing_bounds.pdf).
- 3 Vitalik Buterin. Scalability, part 2: Hypercubes. URL: <https://blog.ethereum.org/2014/10/21/scalability-part-2-hypercubes/>.
- 4 Vitalik Buterin. Minimal viable plasma, 2018. URL: <https://ethresear.ch/t/minimal-viable-plasma>.
- 5 Vitalik Buterin. Plasma cash: Plasma with much less per-user data checking, 2018. URL: <https://ethresear.ch/t/plasma-cash-plasma-with-much-less-per-user-data-checking/1298>.
- 6 Vitalik Buterin. The dawn of hybrid layer 2 protocols. [https://vitalik.ca/general/2019/08/28/hybrid\\_layer\\_2.html](https://vitalik.ca/general/2019/08/28/hybrid_layer_2.html), August 2019. (Accessed on 02/08/2020).
- 7 Christian Cachin and Jan Camenisch. Optimistic fair secure computation. In Mihir Bellare, editor, *Advances in Cryptology - CRYPTO 2000, 20th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 2000, Proceedings*, volume 1880 of *Lecture Notes in Computer Science*, pages 93–111. Springer, 2000. doi:10.1007/3-540-44598-6\_6.
- 8 Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd Annual Symposium on Foundations of Computer Science, FOCS 2001, 14-17 October 2001, Las Vegas, Nevada, USA*, pages 136–145. IEEE Computer Society, 2001. doi:10.1109/SFCS.2001.959888.
- 9 Stefan Dziembowski, Lisa Ekey, and Sebastian Faust. Fairswap: How to fairly exchange digital goods. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018*, pages 967–984. ACM, 2018. doi:10.1145/3243734.3243857.
- 10 Stefan Dziembowski, Lisa Ekey, Sebastian Faust, Julia Hesse, and Kristina Hostáková. Multi-party virtual state channels. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part I*, volume 11476 of *Lecture Notes in Computer Science*, pages 625–656. Springer, 2019. doi:10.1007/978-3-030-17653-2\_21.
- 11 Stefan Dziembowski, Grzegorz Fabiański, Sebastian Faust, and Siavash Riahi. Lower bounds for off-chain protocols: Exploring the limits of plasma. Cryptology ePrint Archive, Report 2020/175, 2020. URL: <https://eprint.iacr.org/2020/175>.
- 12 Stefan Dziembowski, Sebastian Faust, and Kristina Hostáková. General state channel networks. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018*, pages 949–966. ACM, 2018. doi:10.1145/3243734.3243856.



- 13 Georg Fuchsbauer, Michele Orrù, and Yannick Seurin. Aggregate cash systems: A cryptographic investigation of mumblewimble. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part I*, volume 11476 of *Lecture Notes in Computer Science*, pages 657–689. Springer, 2019. doi:10.1007/978-3-030-17653-2\_22.
- 14 Juan A. Garay, Aggelos Kiayias, and Nikos Leonardos. The bitcoin backbone protocol: Analysis and applications. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part II*, volume 9057 of *Lecture Notes in Computer Science*, pages 281–310. Springer, 2015. doi:10.1007/978-3-662-46803-6\_10.
- 15 Lior Goldberg and Oren Katz. Starkdex deep dive: Contracts & statement - starkware - medium. <https://medium.com/starkware/tagged/starkdex-specs>, 2019. (Accessed on 02/08/2020).
- 16 Lewis Gudgeon, Pedro Moreno-Sanchez, Stefanie Roos, Patrick McCorry, and Arthur Gervais. Sok: Layer-two blockchain protocols. In Joseph Bonneau and Nadia Heninger, editors, *Financial Cryptography and Data Security - 24th International Conference, FC 2020, Kota Kinabalu, Malaysia, February 10-14, 2020 Revised Selected Papers*, volume 12059 of *Lecture Notes in Computer Science*, pages 201–226. Springer, 2020. doi:10.1007/978-3-030-51280-4\_12.
- 17 Stuart Haber and W. Scott Stornetta. How to time-stamp a digital document. *J. Cryptology*, 3(2):99–111, 1991. doi:10.1007/BF00196791.
- 18 Russell Impagliazzo and Steven Rudich. Limits on the provable consequences of one-way permutations. In David S. Johnson, editor, *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washington, USA*, pages 44–61. ACM, 1989. doi:10.1145/73007.73012.
- 19 Harry A. Kalodner, Miles Carlsten, Paul Ellenbogen, Joseph Bonneau, and Arvind Narayanan. An empirical study of namecoin and lessons for decentralized namespace design. In *14th Annual Workshop on the Economics of Information Security, WEIS 2015, Delft, The Netherlands, 22-23 June, 2015*, 2015. URL: [http://www.econinfosec.org/archive/weis2015/papers/WEIS\\_2015\\_kalodner.pdf](http://www.econinfosec.org/archive/weis2015/papers/WEIS_2015_kalodner.pdf).
- 20 Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography (Chapman & Hall/Crc Cryptography and Network Security Series)*. Chapman & Hall/CRC, 2007.
- 21 Rami Khalil, Alexei Zamyatin, Guillaume Felley, Pedro Moreno-Sanchez, and Arthur Gervais. Commit-chains: Secure, scalable off-chain payments. *Cryptology ePrint Archive*, Report 2018/642, 2018. URL: <https://eprint.iacr.org/2018/642>.
- 22 Aggelos Kiayias and Orfeas Stefanos Thyfronitis Litos. A composable security treatment of the lightning network. *IACR Cryptology ePrint Archive*, 2019:778, 2019. URL: <https://eprint.iacr.org/2019/778>.
- 23 Georgios Konstantopoulos. Plasma cash: Towards more efficient plasma constructions, 2019. URL: <https://www.gakonst.com/plasmacash.pdf>.
- 24 Rajarshi Mitra. Plasma breakthrough: Omisego (omg) announces the launch of ari. <https://www.fxstreet.com/cryptocurrencies/news/plasma-breakthrough-omisego-omg-announces-the-launch-of-ari-201904120245>. (Accessed on 02/08/2020).
- 25 Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2009. URL: <http://bitcoin.org/bitcoin.pdf>.
- 26 Joseph Poon and Vitalik Buterin. Plasma: Scalable autonomous smart contracts, 2017. URL: <http://plasma.io/plasma.pdf>.
- 27 Nick Szabo. Smart contracts: Building blocks for digital markets. *Extropy Magazine*.
- 28 Trustnodes. Ethereum transactions fall off the cliff, three plasma projects close to release says buterin, 2018. URL: <https://www.trustnodes.com/2018/07/05/ethereum-transactions-fall-off-cliff-three-plasma-projects-close-release-says-buterin>.



## 72:20 Exploring the Limits of Plasma

- 29 Wikipedia. Trusted timestamping. URL: [https://en.wikipedia.org/wiki/Trusted\\_timestamping](https://en.wikipedia.org/wiki/Trusted_timestamping).
- 30 Joon Ian Wong. The ethereum network is getting jammed up because people are rushing to buy cartoon cats on its blockchain. Quartz, 2017. URL: <https://qz.com/1145833/cryptokitties-is-causing-ethereum-network-congestion/>.
- 31 Gavin Wood. Ethereum: A secure decentralised generalised transaction ledger, 2016. URL: <http://gavwood.com/paper.pdf>.

# Majorizing Measures for the Optimizer

**Sander Borst**

Centrum Wiskunde & Informatica, Amsterdam, The Netherlands  
sander.borst@cwi.nl

**Daniel Dadush**

Centrum Wiskunde & Informatica, Amsterdam, The Netherlands  
d.n.dadush@cwi.nl

**Neil Olver**

London School of Economics and Political Science, UK  
n.olver@lse.ac.uk

**Makrand Sinha**

Centrum Wiskunde & Informatica, Amsterdam, The Netherlands  
makrand.sinha@cwi.nl

---

## Abstract

The theory of majorizing measures, extensively developed by Fernique, Talagrand and many others, provides one of the most general frameworks for controlling the behavior of stochastic processes. In particular, it can be applied to derive quantitative bounds on the expected suprema and the degree of continuity of sample paths for many processes.

One of the crowning achievements of the theory is Talagrand's tight alternative characterization of the suprema of Gaussian processes in terms of majorizing measures. The proof of this theorem was difficult, and thus considerable effort was put into the task of developing both shorter and easier to understand proofs. A major reason for this difficulty was considered to be theory of majorizing measures itself, which had the reputation of being opaque and mysterious. As a consequence, most recent treatments of the theory (including by Talagrand himself) have eschewed the use of majorizing measures in favor of a purely combinatorial approach (the *generic chaining*) where objects based on sequences of partitions provide roughly matching upper and lower bounds on the desired expected supremum.

In this paper, we return to majorizing measures as a primary object of study, and give a viewpoint that we think is natural and clarifying from an optimization perspective. As our main contribution, we give an algorithmic proof of the majorizing measures theorem based on two parts:

- We make the simple (but apparently new) observation that finding the best majorizing measure can be cast as a convex program. This also allows for efficiently computing the measure using off-the-shelf methods from convex optimization.
- We obtain tree-based upper and lower bound certificates by *rounding*, in a series of steps, the primal and dual solutions to this convex program.

While duality has conceptually been part of the theory since its beginnings, as far as we are aware no explicit link to convex optimization has been previously made.

**2012 ACM Subject Classification** Mathematics of computing → Stochastic processes; Mathematics of computing → Convex optimization; Theory of computation → Randomness, geometry and discrete structures

**Keywords and phrases** Majorizing measures, Generic chaining, Gaussian processes, Convex optimization, Dimensionality Reduction

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.73

**Related Version** <https://arxiv.org/abs/2012.13306>

**Funding** *Sander Borst*: Supported by the ERC Starting grant QIP-805241.

*Daniel Dadush*: Supported by the ERC Starting grant QIP-805241.

*Neil Olver*: Supported by NWO VIDI grant 016.Vidi.189.087.

*Makrand Sinha*: Supported by the NWO VICI grant 639.023.812.



© Sander Borst, Daniel Dadush, Neil Olver, and Makrand Sinha;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 73; pp. 73:1–73:20



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

## 1 Introduction

Let  $(Z_x)_{x \in X}$  denote a family of centered (mean zero) jointly Gaussian random variables, indexed by points of a set  $X$ . A fundamental statistic of such a process is the expected supremum  $\mathbb{E}[\sup_{x \in X} Z_x]$ , which provides an important measure of the size of the process. This statistic has applications in a wide variety of areas. We list some relevant examples. In convex geometry, one can associate a process to any symmetric convex body  $K$ , whose supremum gives lower bounds on the size of the largest nearly spherical sections of  $K$  [13]. In the context of dimensionality reduction, one can associate a Gaussian process to any point set  $S$  in  $\mathbb{R}^d$  whose squared expected supremum upper bounds the projection dimension needed to approximately preserve distances between points in  $S$  [7, 14]. In the study of Markov Chains, the square of the expected supremum of the Gaussian free field of a graph  $G$  was shown to characterize the cover time of the simple random walk on  $G$  [3].

The above list of applications, which is by no means exhaustive, help motivate the interest in many areas of Mathematics for obtaining a fine grained understanding of such suprema. We now retrace some of the key developments in the theory of Gaussian processes leading up to Talagrand's celebrated majorizing measure theorem [17], which gives an alternate characterization of Gaussian suprema in terms of an optimization problem over measures on  $X$ . The goal of this paper is to give a novel optimization based perspective on this theory, as well as a new *constructive* proof of Talagrand's theorem. For this purpose, some of the earlier concepts, in particular, majorizing measures, will be central to the exposition. We will also cover some generalizations of the theory to the non-Gaussian setting, as our results will be applicable there as well. Throughout our exposition, we rely on the terminology introduced by van Handel [22] for the various combinatorial objects within the theory (i.e., labelled nets, admissible nets and packing trees).

### 1.1 Bounding the Supremum of Stochastic Processes

In what follows we use the notation  $A \lesssim B$  ( $A \gtrsim B$ ) if there exists an absolute constant  $c > 0$  such that  $A \leq cB$  ( $cA \geq B$ ). We use  $A \asymp B$  to denote  $A \lesssim B$  and  $A \gtrsim B$ .

A first basic question one may ask is what information about the Gaussian process  $(Z_x)_{x \in X}$  is sufficient to exactly characterize the expected supremum? An answer to this problem was given by Sudakov [16], strengthening a result of Slepian [15]. Sudakov showed that it is uniquely identified by the natural (pseudo) distance metric

$$d(u, v) := \mathbb{E}[(Z_u - Z_v)^2]^{1/2}, \quad \forall u, v \in X. \quad (1.1)$$

In fact, Sudakov proved the following stronger comparison theorem: if  $(Y_x)_{x \in X}$  and  $(Z_x)_{x \in X}$  are Gaussian processes on the same index set  $X$  and for every  $u, v \in X$ , it holds that  $\mathbb{E}[(Y_u - Y_v)^2] \leq \mathbb{E}[(Z_u - Z_v)^2]$ , then  $\mathbb{E}[\sup_x Y_x] \leq \mathbb{E}[\sup_x Z_x]$ .

Given the above, it is natural to wonder what properties of the metric space  $X$  allow us to obtain upper and lower bounds on  $\mathbb{E}[\sup_{x \in X} Z_x]$ ? A first intuitively relevant quantity is the diameter of  $X$  defined by  $\mathbf{D}(X) := \sup_{u, v \in X} d(u, v)$ . For any  $u, v \in X$ , we have the following simple lower bound:

$$\begin{aligned} \mathbb{E}[\sup_x Z_x] &\geq \mathbb{E}[\max\{Z_u, Z_v\}] = \mathbb{E}[\max\{Z_u - Z_v, 0\}] + \mathbb{E}[Z_v] \\ &= \frac{1}{2} \mathbb{E}[|Z_u - Z_v|] = \frac{d(u, v)}{\sqrt{2\pi}}. \end{aligned} \quad (1.2)$$

Here, we use that  $\mathbb{E}[Z_v] = 0$ , and that  $Z_u - Z_v$  is Gaussian with variance  $d(u, v)^2$ . Thus  $\mathbb{E}[\sup_x Z_x] \geq \mathbf{D}(X)/\sqrt{2\pi}$ .

Instead of looking at two maximally separated points, one might expect to get stronger lower bounds using a large set of well-separated points in  $X$ . Such an inequality was given by Sudakov [16], who showed that

$$\max_{r>0} r \sqrt{\log N_X(r)} \lesssim \mathbb{E}[\sup_x Z_x],$$

where  $N_X(r) := \min\{|S| \mid S \subseteq X, \forall x \in X, \min_{s \in S} d(x, s) \leq r\}$  is the minimum size of an  $r$ -net of  $X$ . This is in fact a direct consequence of Sudakov's comparison theorem. Precisely, the restriction of the process  $\{Z_x\}_{x \in X}$  to a suitable  $r$ -net  $S$ , chosen greedily so that every two points in  $S$  are at distance at least  $r$ , majorizes the maximum of  $|S| \geq N_X(r)$  independent Gaussians with standard deviation  $r/\sqrt{2}$ , where a standard computation then yields the left-hand side.

On the upper bound side, Dudley [4] proved that the covering numbers can in fact be *chained* together to upper bound the supremum:

$$\mathbb{E}[\sup_x Z_x] \lesssim \int_0^\infty \sqrt{\log N_X(r)} dr. \quad (1.3)$$

Note that the integral can be restricted to the range  $r \in (0, \mathbf{D}(X)]$ , since  $\log N_X(\mathbf{D}(X)) = \log 1 = 0$ . Dudley's proof of this inequality was extremely influential and showed the power of combining simple tail bounds on pairs of variables  $Z_u - Z_v$  to get a global bound on the supremum. In particular, the main inequality used in Dudley's proof is the standard Gaussian tail bound: for  $u, v \in X$ , and for any  $s > 0$

$$\mathbb{P}[|Z_u - Z_v| \geq d(u, v) \cdot s] \leq 2e^{-s^2/2}. \quad (1.4)$$

The strategy of combining the above inequalities to control the maximum of a process is what is now called *chaining*.

**Basics of Chaining.** The concept of chaining is central to this paper, so we explain the basic mechanics here. As it will be more convenient for the exposition, we will more directly work with symmetric version of the supremum

$$\sup_{x_1, x_2 \in X} Z_{x_1} - Z_{x_2}$$

which is always non-negative. Note that since  $(Z_x)_{x \in X}$  and  $(-Z_x)_{x \in X}$  are identically distributed,

$$\mathbb{E} \left[ \sup_{x_1, x_2 \in X} Z_{x_1} - Z_{x_2} \right] = \mathbb{E} \left[ \sup_{x \in X} Z_x \right] + \mathbb{E} \left[ \sup_{x \in X} -Z_x \right] = 2\mathbb{E} \left[ \sup_{x \in X} Z_x \right],$$

and thus the expected supremum is the same after dividing by 2.

From here, instead of bounding the expectation, we focus on upper bounding the median of  $\sup_{x_1, x_2 \in X} Z_{x_1} - Z_{x_2}$ , which is known to be within a constant factor of the expectation. Precisely, we seek to compute a number  $M > 0$  such that  $\mathbb{P}[\sup_{x_1, x_2 \in X} Z_{x_1} - Z_{x_2} \geq M] \leq 1/2$ . To arrive at such bounds, we define the notion of a *chaining tree*.

► **Definition 1.1** (Chaining Tree). *A (Gaussian) chaining tree  $\mathcal{C}$  for a finite metric space  $(X, d)$  is a rooted spanning tree on  $X$ , with root node  $w \in X$ , together with probability labels  $p_e \in (0, 1/2)$ , for each edge  $e \in E[\mathcal{C}]$ . The edge probabilities are required to satisfy*

### 73:4 Majorizing Measures for the Optimizer

$\sum_{e \in E[\mathcal{C}]} p_e \leq 1/2$ . For each edge  $\{u, v\} = e \in E[\mathcal{C}]$ , we define the induced edge length  $l_e := l_e(p_e, e)$  to satisfy

$$\mathbb{P}_{Z \in \mathcal{N}(0, d(u, v)^2)}[|Z| \geq l_e] = p_e. \quad (1.5)$$

For each  $x \in X$ , let  $\mathcal{P}_x$  denote the unique path from  $x$  to the root  $w$  in  $\mathcal{C}$ . We define the value of  $\mathcal{C}$  to be

$$\text{val}(\mathcal{C}) = \max_{x \in X} \sum_{e \in \mathcal{P}_x} l_e. \quad (1.6)$$

For a Gaussian process  $(Z_x)_{x \in X}$ , where  $d$  is the induced metric as in (1.1), for any chaining tree  $\mathcal{C}$  on  $X$ , we now show that

$$\mathbb{P} \left[ \sup_{x_1, x_2} Z_{x_1} - Z_{x_2} \geq 2 \cdot \text{val}(\mathcal{C}) \right] \leq 1/2. \quad (1.7)$$

By construction, for any edge  $\{u, v\} \in E[\mathcal{C}]$  we first note that

$$\mathbb{P}[|Z_u - Z_v| \geq l_e] = p_e,$$

recalling that  $Z_u - Z_v$  is distributed as  $\mathcal{N}(0, d(u, v)^2)$ . Since  $\sum_{e \in \mathcal{C}} p_e \leq 1/2$ , by the union bound the event  $\mathcal{E}$  defined as “ $|Z_u - Z_v| \leq l_{u, v}$ ,  $\forall \{u, v\} \in E[\mathcal{C}]$ ”, holds with probability at least  $1/2$ . For  $x \in X$ , let us now define  $\mathcal{P}_x$  to be the unique path from the root  $w$  to  $x$  in  $\mathcal{C}$ .

Conditioning on the event  $\mathcal{E}$ , by the triangle inequality

$$|Z_x - Z_w| \leq \sum_{\{u, v\} \in \mathcal{P}_x} |Z_u - Z_v| \leq \sum_{e \in \mathcal{P}_x} l_e. \quad (1.8)$$

Applying the triangle inequality again, we have that

$$\sup_{x_1, x_2} Z_{x_1} - Z_{x_2} \leq 2 \sup_{x \in X} |Z_x - Z_w| \leq 2 \cdot \text{val}(\mathcal{C}).$$

The bound (1.7) now follows since the above occurs with probability at least  $1/2$ .

To work with such chaining trees, it is important to have easy approximations of the edge lengths used above. For  $e = \{u, v\} \in E[\mathcal{C}]$  and  $p_e \in (0, 1/2]$ , it is well known that

$$l_e \asymp d(u, v) \sqrt{\log(1/p_e)}. \quad (1.9)$$

Note that by the standard Gaussian tail bounds (1.4), for any  $p \in (0, 1)$ , we have the upper bound,  $l_e \leq d(u, v) \sqrt{2 \log(2/p_e)}$ .

To relate to earlier lower bounds, it is instructive to see that  $\text{val}(\mathcal{C}) \gtrsim \mathbf{D}(X)$ , for any chaining tree. Firstly, since each  $p_{u, v} \in (0, 1/2]$ , by (1.9), it follows that  $l_{u, v} \gtrsim d(u, v)$ . From here, for any pair of points  $u, v \in X$ , by the triangle inequality  $2 \cdot \text{val}(\mathcal{C})$  pays for the cost of going from  $u$  to the root  $w$  and from  $w$  to  $v$ , yielding the desired upper bound on the diameter.

**Chaining beyond Gaussians.** Importantly, in the above framework, the only element specific to Gaussian processes is the edge length function (1.5). As our results will apply to this more general setting, we explain how chaining can straightforwardly be adapted to work with processes satisfying appropriate tail bounds.

Let us examine a jointly distributed sequence of random variables  $(Z_x)_{x \in X}$  indexed by a metric space  $(X, d)$ . To constrain the process we will make the following assumptions on the tails. Let  $f : \mathbb{R}_+ \rightarrow \mathbb{R}_+$  be a continuous and non-increasing probability density function on the non-negative reals and let  $F(s) = \int_s^\infty f(t) dt$  denote the complementary cumulative distribution function. Then, for all  $x_1, x_2 \in X$  and  $s \geq 0$ , we assume that

$$\mathbb{P}[|Z_{x_1} - Z_{x_2}| \geq d(x_1, x_2) \cdot s] \leq F(s). \quad (1.10)$$

► **Definition 1.2** (Chaining Functional). *We define the chaining functional induced by  $f$  to be  $h(p) := h_f(p) = F^{-1}(p)$ , for  $p \in (0, 1]$ , which is well-defined since  $f$  is non-increasing. Note that  $h(1) = 0$  and that  $h(p)$  is strictly decreasing on  $(0, 1]$ . We say that  $h$  is of log-concave type if the density  $f$  is log-concave.*

A property that we make crucial use of is that  $h(p)$  is, in fact, a convex function of  $p \in (0, 1]$ . To see this, for  $p \in (0, 1)$ , a direct computation yields that  $h'(p) = 1/F'(h(p)) = -1/f(h(p))$ , where the derivative is well-defined since  $f$  is continuous and non-decreasing. Since  $f(s)$  is non-increasing and  $h(p)$  is strictly decreasing,  $h'(p)$  is non-decreasing and hence,  $h$  is convex. Throughout the rest of the paper, we will mainly be interested in chaining functionals of log-concave type.

To apply the chaining framework to the process  $(Z_x)_{x \in X}$ , we use a chaining tree  $\mathcal{C}$  exactly as in Definition 1.1 except that we now compute the edge lengths according to the chaining functional  $h$ . Specifically, for  $e = \{u, v\} \in E[\mathcal{C}]$  and probability  $p_e \in (0, 1)$ , we define

$$l_e := l_e(e, p_e) := d(u, v) \cdot h(p_e). \quad (1.11)$$

We now define  $\text{val}_h(\mathcal{C})$  exactly as in (1.6), using  $h$  to compute the edge lengths.

With this setup, with an identical proof to the previous section, we have the inequality

$$\mathbb{P} \left[ \sup_{x_1, x_2 \in X} Z_{x_1} - Z_{x_2} \geq 2 \cdot \text{val}_h(\mathcal{C}) \right] \leq \frac{1}{2}.$$

As in the Gaussian setup, it is useful to keep in mind what the “trivial” diameter lower bound on  $\text{val}_h(\mathcal{C})$  should be. Since the edge probability  $p_e \in (0, 1/2]$ , for  $e = \{u, v\} \in E[\mathcal{C}]$ , we have that  $l_e \geq d(u, v) \cdot h(1/2)$ . Therefore, for any  $u, v \in X$ , by the triangle inequality, the cost of the paths from  $u$  or  $v$  to the root  $w$  is at least  $d(u, v) \cdot h(1/2)$  for any chaining tree  $\mathcal{T}$ . In particular, for any chaining tree  $\mathcal{C}$ , we derive the lower bound

$$2 \cdot \text{val}_h(\mathcal{C}) \geq \mathbf{D}(X) \cdot h(1/2). \quad (1.12)$$

It is important to note that the Gaussian chaining setup is indeed a special case of the above. Precisely, in that setup the edge lengths are  $l_{u,v} := d(u, v) \cdot h_f(p_{u,v})$ , where  $f(s) = \sqrt{\frac{2}{\pi}} e^{-s^2/2}$  is the density of the absolute value of the standard Gaussian.

**Dudley’s Construction.** To gain intuition about how to apply the chaining framework, we now explain how to build and analyze the chaining tree used in Dudley’s inequality. For simplicity of notation, let us assume that the diameter  $\mathbf{D}(X) = 1$ . For each  $k \geq 0$ , let  $\mathcal{N}_k$  denote a  $2^{-k}$ -net of  $X$  of minimum size, i.e., satisfying  $|\mathcal{N}_k| = N_X(2^{-k})$ . By our diameter assumption,  $\mathcal{N}_0 = \{w\}$  is clearly a single point, which gives the root of the tree  $\mathcal{C}$ . From here, we construct the tree by induction on  $k \geq 1$ . At iteration  $k \geq 1$ , we attach each element of  $\mathcal{N}_k$  not already in  $\mathcal{C}$  to a closest point in  $\mathcal{C}$ . From here, we set the edge probability  $p_{u,v} = p_k := 2^{-(k+1)}/|\mathcal{N}_k|$  and let  $l_{u,v} > 0$  be minimal subject to  $\mathbb{P}[|Z_u - Z_v| \geq l_{u,v}] \leq p_k$ . This completes the construction.

To analyze the tree  $\mathcal{C}$ , we make the following observations. Firstly, the number of edges we add to the tree at iteration  $k$  is at most  $|\mathcal{N}_k|$ . Therefore, the total probability sum is at most  $\sum_{k=1}^{\infty} |\mathcal{N}_k| \cdot p_k = 1/2$ , and hence  $\mathcal{C}$  is a valid chaining tree. Second, any edge  $\{u, v\}$  added during iteration  $k$  satisfies  $d(u, v) \leq 2^{-k+1}$ . Consequently, by the Gaussian tail bound (1.4),

$$l_{u,v} \lesssim d(u, v) \sqrt{\log(1/p_k)} \lesssim 2^{-k+1} \left( \sqrt{\log N_X(2^{-k})} + \sqrt{k+1} \right).$$

In particular, the value of  $\mathcal{C}$  satisfies

$$\text{val}(\mathcal{C}) \lesssim \sum_{k=1}^{\infty} 2^{-k+1} \left( \sqrt{\log N_X(2^{-k})} + \sqrt{k+1} \right) \lesssim 1 + \sum_{k=1}^{\infty} 2^{-k+1} \sqrt{\log(N_X(2^{-k}))}.$$

One can now easily show that the above expression is upper bounded by (1.3) by discretizing the range of the integral along powers of 2 (recalling that  $\mathbf{D}(X) = 1$ ).

**The Method of Majorizing Measures.** Given the above, it is natural to wonder how one might construct an *optimal* chaining tree for a given process  $(Z_x)_{x \in X}$ . A principal goal of this paper will be to give efficient constructions for such trees. At first sight, this may seem like a daunting task, as one must somehow simultaneously optimize over all spanning trees and edge probabilities. Nevertheless, a major step towards this goal was achieved for Gaussian processes by Fernique [5], who proved the following remarkable theorem:

$$\mathbb{E} \left[ \sup_{x \in X} Z_x \right] \lesssim \gamma_2(X) := \inf_{\mu} \sup_{x \in X} \int_0^{\infty} g(\mu(B(x, r))) dr. \quad (1.13)$$

Some definitions are in order. Firstly, the infimum over  $\mu$  is taken over all probability measures on  $X$ . Secondly,  $g(p) := \sqrt{\log(1/p)}$ , for  $p \in [0, 1]$  corresponds to (an approximation of) the Gaussian edge length function in (1.5). Lastly,  $B(x, r) = \{y \in X : d(x, y) \leq r\}$  is the metric ball of radius  $r$  around  $x$ , where  $d$  is the canonical metric induced by the Gaussian process.

Importantly, the natural analogue of  $\gamma_2$  for the general setup in (1.10) also yields upper bounds on the expected supremum, provided the tails of  $f$  decay sufficiently quickly. In particular, for  $(Z_x)_{x \in X}$  satisfying (1.10), for any “nice enough”  $f$ , we have that

$$\mathbb{E} \left[ \sup_{x_1, x_2 \in X} Z_{x_1} - Z_{x_2} \right] \lesssim \gamma_h(X) := \inf_{\mu} \sup_{x \in X} \int_0^{\infty} h(\mu(B(x, r))) dr, \quad (1.14)$$

where  $h$  is as in (1.11). Note that since  $h(1) = 0$ , one can truncate the range of the integral to  $r \in [0, \mathbf{D}(X)]$ . Very general results of the above type can be found in [18, 1]. We note that the requirements of the process in these works are parametrized in a somewhat different way in terms of Orlicz norms. In this work, we will focus on the setting where the chaining functional  $h$  is of log-concave type (where the tail density  $f$  is log-concave), where these different parametrizations are equivalent. Prototypical examples in this class are the tail densities of *exponential type*, which are proportional to  $e^{-x^q}$ ,  $x \geq 0$ , for  $q \geq 1$ , and where  $h(p) \asymp \ln^{1/q}(1/p)$  for  $p \in (0, 1/2)$ .

Given that any probability measure  $\mu$  can be used to upper bound the expected supremum, Fernique dubbed the above technique the method of *majorizing measures*. It is worthwhile to note that Fernique did not prove inequality (1.13) via chaining. He relied instead on a more general technique, which first proves a generic concentration inequality for real valued functions on the metric space, and recovers the desired inequality by averaging over the ensemble of functions induced by the process. The fact that one can recover the same bound via chaining for Gaussian processes would only be proved later, at first implicitly in Talagrand [17], and explicitly in [21], where the latter work also covered processes of exponential type mentioned above.

As noted above, the quantity  $\gamma_2(X)$  and more generally  $\gamma_h(X)$  (for  $h$  of log-concave type), rather miraculously models the value of the best chaining tree as a continuous optimization problem. As majorizing measures may seem like rather opaque objects at first sight, we



believe it is instructive to note that from a chaining tree  $\mathcal{C}$ , one can construct a measure  $\mu$  whose value in (1.13) is at most  $3 \cdot \text{val}_h(\mathcal{C})$ . The construction is simple: set  $\mu_w = 1/2$  on the root  $w$ , and for each  $e = \{u, v\} \in E[\mathcal{C}]$  (with  $v$  closer to the root than  $u$ ), set  $\mu_u = p_e$ . The details of the comparison can be found in the appendix of the full version of the paper.

From the above discussion, we see that the majorizing measures are indeed powerful tools for upper bounding suprema. Given this, together with the many tools for *lower bounding* Gaussian suprema (which are not available in general), Fernique [5] conjectured that majorizing measures should fully characterize the expected supremum of Gaussian processes. This conjecture was verified in the ground-breaking work of Talagrand [17], which is now called the majorizing measures theorem:

► **Theorem 1.3** (Fernique-Talagrand [5, 17]). *For any centered Gaussian process  $(Z_x)_{x \in X}$  over the metric space  $X = (X, d)$ , where  $d$  is the canonical metric induced by the process, we have*

$$\mathbb{E} \left[ \sup_{x \in X} Z_x \right] \asymp \gamma_2(X).$$

The original proof of the majorizing measure theorem [17] was considered notoriously difficult. Due to its importance in the theory of stochastic processes, many simpler as well as different proofs were found [19, 20, 21, 12, 2, 22], often by Talagrand himself.

As stated at the beginning of the introduction, the goal of this paper is to give an alternative constructive proof of this theorem using a convex optimization approach. In particular, our starting point is the simple observation that  $\gamma_h(X)$  is in fact a convex program, which follows directly from the convexity of  $h^1$ . While simple (and most certainly known to experts), we have not seen this observation leveraged in earlier proofs. In our context, convexity will allow for near-optimal solutions to  $\gamma_h(X)$  to be efficiently computed using off-the-shelf methods. Furthermore, convex duality will allow us to inspect the structure of solutions to natural dual program(s) for  $\gamma_h(X)$ , enabling us to reason about lower bounds. Our proof will operate entirely at the level of the metric space, and will produce a natural combinatorial variant of an optimal primal-dual solution pair for  $\gamma_h(X)$ , namely a chaining tree and packing tree (defined shortly). These solutions will in fact be obtained by “rounding” solutions to the corresponding continuous programs. Specializing to the Gaussian case, we recover the majorizing measure theorem by an easy comparison between the value of the Gaussian supremum and the value of the combinatorial solutions (which are tailor made for this purpose). This strategy has the benefit of clearly separating the role of the metric space and the role of the Gaussian process, which are often intertwined in difficult to disentangle ways in many proofs.

We now review some of the key ideas in known proofs, which will be important for our approach as well. In particular, we will require appropriate dual analogue to chaining trees.

**Primal Proof Strategies.** Given what we have seen so far, a main missing ingredient is a stronger form of lower bound for the value of the Gaussian supremum (noting that chaining already provides the upper bound). For this purpose, we examine the natural functional induced by the process on subsets of  $X$ , defining

$$G(S) := \mathbb{E} \left[ \sup_{x \in S} Z_x \right], \quad \forall S \subseteq X. \quad (1.15)$$

<sup>1</sup> The formulation  $\gamma_2(X)$  is “essentially convex”. This is because  $g(p)$  is only convex on the interval  $[0, 1/\sqrt{e}]$ , which is easily remedied. We note this non-convexity is principally due to  $g(p)$  being a poor approximation of (1.5) for  $p \in [1/\sqrt{e}, 1]$ .

The following functional inequality, named the “super-Sudakov” inequality in [22], was proven in [19]: there exists  $\gamma \in (0, 1)$ , such that given an  $r$ -separated (non-empty) subsets  $A_1, \dots, A_N \subseteq S$ , i.e., satisfying  $d(A_i, A_j) \geq r$ ,  $\forall i \neq j$ , and  $\mathbf{D}(A_i) \leq \gamma r$ ,  $\forall i \in [N]$ , then

$$G(S) \geq \gamma \cdot r \cdot g(1/N) + \min_{i \in [N]} G(A_i), \quad (1.16)$$

where  $g(x) = \sqrt{\log 1/x}$  for  $x \in [0, 1]$ , as before.

In [19], Talagrand gave a construction which takes a functional  $G$  on  $X$  satisfying (1.16), and produces (a variant of) a chaining tree  $\mathcal{C}$  satisfying  $\text{val}(\mathcal{C}) \lesssim G(X)$ . Talagrand’s construction is based on a recursive partitioning scheme, where the partitions roughly correspond to subtrees, which greedily chooses metric balls of large  $G$  value to construct the partition. We note that this construction comes in different flavors, each yielding more structured versions of chaining trees (i.e., labelled nets [19] and admissible nets [20]). By instantiating  $G$  to be the functional given by (1.15) immediately yields Theorem 1.3. While Talagrand’s construction was certainly algorithmic, the Gaussian functional  $G$  is not easy to compute (at least deterministically). As mentioned previously, in [21], Talagrand also gave another procedure that directly converts measures to chaining trees. Note that this yields a good chaining tree from a good measure, but by itself does not yield Theorem 1.3.

**Dual Proof Strategies.** One reason the “difficult” Gaussian functional  $G$  was required to prove Theorem 1.3 is that there was no simple dual object to compare to that certifies a lower bound. From the convex optimization perspective, this should morally correspond to a solution to the dual of  $\gamma_2(X)$  (or  $\gamma_h(X)$ ). Such an object, called a *packing tree* in the terminology of [22], was in fact developed in Talagrand’s original proof [17] for the Gaussian case, and extended to general chaining functionals in [10].

► **Definition 1.4 (Packing Tree).** Let  $\alpha \in (0, \frac{1}{10}]$ . An  $\alpha$ -packing tree  $\mathcal{T}$  on a finite metric space  $(X, d)$  is a rooted tree on subsets of  $X$ , with root node  $W \subseteq X$ , together with a labelling  $\chi : \mathcal{T} \rightarrow \mathbb{Z}_{\geq 0}$ . We enforce that every leaf node  $V \in \mathcal{T}$  is a singleton, i.e.,  $V = \{x\}$  for some  $x \in X$ . We denote  $\text{leaf}(\mathcal{T}) \subseteq X$  to be the union of all leaf nodes of  $\mathcal{T}$ . Every node  $V \in \mathcal{T}$  has a (possibly empty) set of children  $C_1, \dots, C_k \subseteq V$  which are pairwise disjoint. We let  $\deg_+(V) := k$  denote the number of children of  $V$ . We enforce the follow metric properties on  $\mathcal{T}$ :

1. For any child  $C$  of  $V \in \mathcal{T}$ , we have that  $\mathbf{D}(C) \leq \alpha^{\chi(V)+1} \cdot \mathbf{D}(X)$ .
2. For  $V \in \mathcal{T}$  and distinct children  $C_1, C_2$  of  $V$ , we have  $d(C_1, C_2) \geq \frac{1}{10} \alpha^{\chi(V)} \cdot \mathbf{D}(X)$ .

The value of an  $\alpha$ -packing tree  $\mathcal{T}$  with respect to a chaining functional  $h$  is defined as

$$\text{val}_h(\mathcal{T}) := \inf_{x \in \text{leaf}(\mathcal{T})} \sum_{V \in \mathcal{P}_x \setminus \{x\}} \alpha^{\chi(V)} \cdot \mathbf{D}(X) \cdot h(1/\deg_+(V)), \quad (1.17)$$

where  $\mathcal{P}_x$  is the unique path from the root  $W$  to the leaf  $\{x\}$ . We use the shorthand  $\text{val}_2(\mathcal{T})$  to denote the value with respect to the Gaussian functional  $g$ .

We remark that we do not count the edge going to the parent in  $\deg_+(V)$  mostly for notational convenience – in this case, nodes with a sole child do not contribute to the value of the packing tree. Also, there is quite a bit of flexibility in the parameters of the packing tree, which are chosen above for convenience. Packing trees are objects that allow us to chain lower bounds together in analogy to upper bounds via chaining trees. The combinatorial structure of a packing tree is more constrained than that of a chaining tree however, and their construction (at least more from the perspective of the analysis) is more delicate.

In the Gaussian setting, an  $\alpha$ -packing tree  $\mathcal{T}$  is perfectly tailored for combining the “super-Sudakov” inequalities given by (1.16). In particular, for  $\alpha = 1/(2\gamma)$ , a direct proof by induction starting from the leaves of the tree certifies that  $G(X) \gtrsim \text{val}_2(\mathcal{T})$  (see Theorem 6.36 in [22]). This was in fact first established in [17] using Slepian’s lemma instead of (1.16). Independently of any process however, they also directly serve as combinatorial lower bounds for  $\gamma_h(X)$ .

► **Lemma 1.5.** *Let  $\alpha \in (0, \frac{1}{10}]$ . For a finite metric space  $(X, d)$ , an  $\alpha$ -packing tree  $\mathcal{T}$  on  $X$ , and any chaining functional  $h$ , we have*

$$\gamma_h(X) \geq \frac{1}{2}(1 - \alpha) \cdot \text{val}_h(\mathcal{T}).$$

While known to experts, it is not so easy to find combinatorial proofs of the above inequality, i.e. not related to a process, in the literature (see for example Exercise 6.12 in [22] or Lemma 3.7 in [3]). We include a proof in the appendix of the full version of the paper.

Talagrand’s original proof of the majorizing measures theorem worked almost entirely on the dual side. As generalized in [10], the main work in the proof was in fact to construct an  $\alpha$ -packing tree  $\mathcal{T}$  satisfying  $\text{val}_h(\mathcal{T}) \gtrsim \gamma_h(\mathcal{T})$  (for  $h$  of log-concave type). As for the primal side, the construction is based on similar greedy ball (sub-)partitioning using an appropriate functional  $H$  on  $X$  satisfying a so-called “super-chaining” inequality in the terminology of [22]. Specifically, for any set  $S \subseteq X$ , and a partition  $S = \sqcup_{i=1}^N P_i$ ,  $H$  satisfies

$$H(S) \leq \max_{i \in [N]} \beta \cdot \mathbf{D}(S)h(1/(i+1)) + H(P_i), \quad (1.18)$$

for some absolute constant  $\beta > 0$ . Interestingly, the functional  $H$  used in [17, 10] was a variant of  $\gamma_h(X)$ , which is deterministically computable, and not the Gaussian functional in the case  $h = g$  (though this works as well [22]). This construction was in fact leveraged in [3] to give a deterministic polynomial time dynamic programming algorithm for computing a nearly optimal packing tree.

## 1.2 Our Results

### 1.2.1 A Constructive Min-Max Theorem

The main result of this paper is the following constructive variant of the combinatorial core of the majorizing measure theorem.

► **Theorem 1.6.** *Let  $(X, d)$  be an  $n$  point metric space,  $h$  be a chaining functional of log-concave type. Then there is a deterministic algorithm which computes a chaining tree  $\mathcal{C}^*$  and an  $1/10$ -packing tree  $\mathcal{T}^*$  satisfying*

$$\text{val}_h(\mathcal{C}^*) \asymp \text{val}_h(\mathcal{T}^*),$$

*using  $\tilde{O}(n^{\omega+1})$  arithmetic operations and evaluations of  $h$  and  $h'$ , where  $\omega \leq 2.373$  is the matrix multiplication constant.*

We note that the packing tree parameter  $1/10$  can be made smaller at the cost increasing the hidden constant in the  $\asymp$  notation. Recall that for any pair of trees  $\mathcal{C}$  and  $\mathcal{T}$  as above, we have already seen that

$$\text{val}_h(\mathcal{C}) \gtrsim \gamma_h(X) \gtrsim \text{val}_h(\mathcal{T}), \quad (1.19)$$

so the pair in Theorem 1.6 form a nearly-optimal primal-dual pair. Furthermore, in the Gaussian setting where  $h = g$ , replacing  $\gamma_2(X)$  above by  $\mathbb{E}[\sup_x Z_x]$  corresponds to the “easy direction” of the majorizing measures theorem. Plugging in the solutions from Theorem 1.6 immediately yield the hard direction of the theorem. This allows us to view the metric space part of the majorizing measure theorem as an instance of a combinatorial min-max theorem. We remark that such a combinatorial min-max characterization of the Majorizing Measures theorem was already observed by Guédon and Zvavitch [8]. They showed that the value of the optimal packing tree defines a functional that satisfies the super-Sudakov inequality; when combined with Talagrand’s framework, this implies the combinatorial min-max theorem described above. This does not directly yield deterministic constructions of nearly-optimal chaining or packing trees, however.

Ding, Lee and Peres [3] essentially used this observation of [8] along with a suitable dynamic program to give an efficient deterministic algorithm to compute nearly optimal packing trees, as mentioned previously. For nearly optimal chaining trees, we make the simple observation (which seems to have gone unnoticed) that these can be extracted from Talagrand’s [21] “rounding” algorithm applied to a nearly optimal solution for the efficiently solvable convex program  $\gamma_h(X)$ . By themselves however, these algorithms do not directly say much about how the values of these different trees relate to each other.

In Theorem 1.6, we build further on the convex programming approach. At a high level, we build the primal and dual solutions at the same time and rely on convex programming duality to ensure they have (nearly) the same value. In essence, we replace the “magic functionals” satisfying super-Sudakov or super-chaining inequalities that appear in Talagrand’s constructions with convex duality. As we will see in the next section, the dual objects will also correspond to probability measures. In contrast to the primal however, where the rounding to a suitable chaining tree can be done in one shot, the dual measures will require multiple levels of rounding.

The primal and dual solutions we require correspond to nearly optimal primal and dual measures associated with a saddle-point formulation of  $\gamma_h(X)$  (see (1.20) in the next subsection). There are in fact many existing solvers that are able to compute nearly optimal solutions to such saddle point problems, where we will rely on a recent fast solver of [9]. This computation in fact forms the bulk of the running time of the algorithm in Theorem 1.6. The details of this part of the algorithm can be found in the full version of this paper. An interesting open problem is whether one can reduce the running time of Theorem 1.6 to  $\tilde{O}(n^2)$ , which would be nearly-linear in the input size (recall that an  $n$  point metric consists of  $n^2$  distances). The main bottleneck is the use of an all purpose blackbox solver [9] to approximately solve (1.20), and it seems likely that an appropriately tailored first-order method could bring the running time down to  $\tilde{O}(n^2)$ .

While our main contribution is conceptual, we expect and hope that novel and interesting applications of an “algorithmic” theory of chaining will be found. As a contribution on this front, we give an application of Theorem 1.6 in the context of derandomization: we give a deterministic algorithm for computing Johnson-Lindenstrauss projections achieving the guarantees of Gordon’s theorem [7], where we rely on the chaining based proof from [14]. As far as we are aware, no prior deterministic construction was known.

### 1.2.2 Simplifying the Dual of $\gamma_h(X)$

For simplicity of notation, throughout this section (and most of the paper), we will assume that  $(X, d)$  is a fixed  $n$ -point metric space and that  $h$  is a chaining functional of log-concave type satisfying  $|h'(1)| = 1$  (interpreted as the left directional derivative). Under this normalization

on  $h$ , the trivial diameter lower bound on  $\gamma_h(X)$  will be at least  $\mathbf{D}(X)/4$ , which we will use to convert additive errors to multiplicative ones. This normalization is without loss of generality, and can be achieved by appropriately scaling the  $h$  and the metric  $d$  so that  $\gamma_h(X)$  remains unchanged (see Section 2.1 for a full explanation).

We now describe the dual formulation of  $\gamma_h(X)$  and describe the process of simplifying it. For this purpose, we start with the basic saddle-point formulation of  $\gamma_h(X)$ :

$$\gamma_h(X) = \min_{\mu} \max_{x \in X} \int_0^\infty h(\mu(B(x, r))) dr = \min_{\mu} \max_{\nu} \int_X \int_0^\infty h(\mu(B(x, r))) dr d\nu(x) \quad (1.20)$$

where  $\nu$  also ranges over all probability measures on  $X$  (the optimal  $\nu$  above puts mass 1 on any maximizer of  $\int_0^\infty h(\mu(B(x, r))) dr$ ).

To obtain the dual program to  $\gamma_h(X)$ , we interchange  $\mu$  and  $\nu$ :

$$\gamma_h(X) \geq \max_{\nu} \min_{\mu} \int_X \int_0^\infty h(\mu(B(x, r))) dr d\nu(x) := \gamma_h^*(X). \quad (1.21)$$

In particular, for any fixed dual measure  $\nu$ , we have

$$\gamma_h(X) \geq \min_{\mu} \int_X \int_0^\infty h(\mu(B(x, r))) dr d\nu(x). \quad (1.22)$$

Since the objective  $\int_X \int_0^\infty h(\mu(B(x, r))) dr d\nu(x)$  is convex in  $\mu$  and linear in  $\nu$ , and the probability simplex is compact and convex, by Sion's theorem the value of both convex programs is equal. That is,  $\gamma_h(X) = \gamma_h^*(X)$ . The measures required within the construction in Theorem 1.6 will be nearly optimal primal and dual measures  $\mu^*$  and  $\nu^*$  to  $\gamma_h(X)$  and  $\gamma_h^*(X)$  respectively.

Unfortunately, it is not clear at this point that the dual is terribly useful. In particular, even evaluating the objective in (1.22) for a given dual measure  $\nu$  requires solving a non-trivial convex optimization problem (note that the corresponding objective of  $\gamma_h(X)$  can be computed by simply evaluating  $n$  integrals). Rather surprisingly, it turns out that for  $h$  of log-concave type, one can in fact “guess” a near-optimal  $\mu$  in (1.22), namely, we can set  $\mu = \nu$ .

► **Lemma 1.7.** *For any probability measure  $\nu$  on  $X$ , we have that*

$$\int_X \int_0^\infty h(\nu(B(x, r))) dr d\nu(x) \leq 2 \min_{\mu} \int_X \int_0^\infty h(\mu(B(x, r))) dr d\nu(x) + \mathbf{D}(X)/e,$$

where the minimum is taken over all probability measures  $\mu$ .

The proof of the above proceeds on a “per scale” basis. More precisely, for a given  $r > 0$ , we show that  $\int_X h(\nu(B(x, 2r))) d\nu(x) \leq \int_X h(\mu(B(x, r))) d\nu(x) + 1/e$ . This statement is easily restated in graph-theoretic terms, by defining a graph  $G = (X_1 \cup X_2, E)$ , with  $X_1, X_2$  both being copies of  $X$  and where  $x_1 \in X_1$  is adjacent to  $x_2 \in X_2$  if  $d(x_1, x_2) \leq r$ . Then  $\mu(B(x, r))$  corresponds to the mass under  $\mu$  within the neighborhood of  $x$ , and  $\nu(B(x, 2r))$  to the mass under  $\nu$  within the two-hop neighborhood of  $x$ . In this setting, we use a tool from combinatorial optimization, namely, a generalization of Hall's theorem. We note the two properties needed from  $h$  above are that  $h$  be decreasing and  $\max_{a \in (0, 1]} |ah'(a)| \leq 1$ . The latter property in fact follows from  $h$  being of log-concave type and the normalization  $|h'(1)| = 1$ .

Motivated by the above, we consider the following simplification of  $\gamma_h^*(X)$ , which we call the *entropic dual*:

$$\delta_h^{\text{Ent}}(X) := \max_{\nu} \int_X \int_0^\infty h(\nu(B(x, r))) dr d\nu(x). \quad (1.23)$$

This corresponds to the value  $\nu$  using the nearly optimal guess for  $\mu$  in (1.22), which is now readily computable. The following direct corollary relates the value of the entropic dual to the actual dual.

► **Corollary 1.8.**

$$\gamma_h^*(X) \leq \delta_h^{\text{Ent}}(X) \leq 2\gamma_h^*(X) + \mathbf{D}(X)/e.$$

In terms of the additive error above, as mentioned at the beginning of the section,  $\mathbf{D}(X)/4$  will be the trivial lower bound on  $\gamma_h^*(X) = \gamma_h(X)$ . Therefore, the right hand in Theorem 1.8 is at most  $(2 + 4/e)\gamma_h^*(X)$  in the worst-case. In most interesting cases however, one would expect  $\gamma_h(X)$  to be far from the trivial lower bound, in which case one can think of the right hand side as  $(2 + o(1))\gamma_h^*(X)$ .

We are now ready to give the final simplified form of the dual whose value will most directly relate to the value of packing trees: we define the *simplified dual* by

$$\delta_h(X) := \max_{\nu} \min_{x \in X, \nu(x) > 0} \int_0^\infty h(\nu(B(x, r))) dr. \quad (1.24)$$

Note that we restrict to the minimum of the points supported by  $\nu$ . These will correspond to the potential leaf nodes in the packing tree. Furthermore, the minimum in (1.24) is in direct analogy to the minimum cost of a path down a packing tree.

Trivially, since we replaced the average by a minimum, we have that  $\delta_h^{\text{Ent}}(X) \geq \delta_h(X)$ . We show that the reverse direction also holds up to additive error. For a probability measure  $\nu$  on  $X$  and for any subset  $S \subseteq X$ , satisfying  $\nu(S) > 0$ , define  $\nu_S$  by

$$\nu_S(A) := \nu_S(A \cap S)/\nu(S), \forall A \subseteq X, \quad (1.25)$$

i.e.,  $\nu_S$  is the conditional probability measure induced by  $\nu$  on  $S$ . The following lemma shows that one can easily convert a measure  $\nu$  with large  $\delta_h^{\text{Ent}}$  value to one with large  $\delta_h$  value via conditioning.

► **Lemma 1.9.** *For any probability measure  $\nu$  on  $X$ , there exists  $S \subseteq \{x \in X : \nu(x) > 0\}$  such that*

$$\int_X \int_0^\infty h(\nu(B(x, r))) dr \leq \min_{x \in S} \int_0^\infty h(\nu_S(B(x, r))) dr + \mathbf{D}(X).$$

Furthermore,  $S$  can be computed using at most  $O(n^3)$  arithmetic operations and evaluations of  $h$ .

The algorithm achieving the above is in fact very simple: we start with  $S = \{x \in X : \nu(x) > 0\}$ , and iteratively kick out the element  $x \in S$  with lowest value as long as the above inequality is not met.

Combining Corollary 1.8 and Lemma 1.9, we obtain the following relations between the dual program  $\gamma_h^*(X)$  and the simplified dual  $\delta_h(X)$ .

► **Theorem 1.10.**

$$\gamma_h^*(X) - \mathbf{D}(X) \leq \delta_h(X) \leq 2\gamma_h^*(X) + \mathbf{D}(X)/e.$$

Furthermore, given any probability measure  $\nu$  on  $X$ , one can compute  $S \subseteq \{x \in X : \nu(x) > 0\}$  satisfying

$$\int_X \int_0^\infty h(\nu(B(x, r))) dr d\nu(x) \leq \min_{x \in S} \int_0^\infty h(\nu_S(B(x, r))) dr + \mathbf{D}(X),$$

using at most  $O(n^3)$  arithmetic operations and evaluations of  $g$ .

We are in fact not the first to examine the  $\delta_h^{\text{Ent}}(X)$  and  $\delta_h(X)$  programs. The analysis of solutions to  $\delta_2^{\text{Ent}}(X)$  (i.e.,  $h = g$ ) in fact already goes back all the way to Fernique [5]. Starting from a Gaussian process  $(Z_x)_{x \in X}$ , Fernique examined the measure  $\nu$  on  $X$  satisfying  $\nu(u) = \mathbb{P}[Z_u = \max_{x \in X} Z_x]$ ,  $\forall u \in X$ , where we assume  $X$  is finite and that the maximum is uniquely attained with probability 1. For this “argmax” measure  $\nu$ , it was shown that

$$\mathbb{E} \left[ \sup_{x \in X} Z_x \right] \asymp \int_X \int_0^\infty g(\nu(B(x, r))) dr d\nu(x),$$

where the inequalities  $\lesssim$  and  $\gtrsim$  were proven by Fernique [5] and Talagrand [17] respectively.

The relationship between  $\gamma_h(X)$  and  $\delta_h(X)$  was also already studied by Naor and Mendel [11] as well as Bednorz [2]. In particular, for any continuous  $h$  satisfying  $\lim_{x \rightarrow 0^+} h(x) = \infty$  (i.e., not necessarily of log-concave type), [11, 2] showed that  $\gamma_h(X) \leq \delta_h(X)$ . This was proved using Brouwer’s fixed-point theorem, which was used to find a measure  $\mu$  where the quantities  $\int_0^\infty h(\mu(B(x, r))) dr$  are equal for all  $x \in X$ . Recalling that  $\gamma_h(X) = \gamma_h^*(X)$ , this bound is stronger than  $\gamma_h^*(X) - \mathbf{D}(X) \leq \delta_h(X)$  in Theorem 1.10. However, we do not require  $\lim_{x \rightarrow 0^+} h(x) = \infty$  (e.g.,  $h(x) = 1 - x$  is valid for us), which is crucially used to prove the existence of the above Brouwer measure. Mendel and Naor [11] further prove that  $\delta_h(X) \lesssim \gamma_h(X)$ , for any decreasing  $h$  satisfying  $h(x^2) \lesssim h(x)$ . This is achieved by rounding any dual measure  $\nu$  to what they call an *ultrametric skeleton*, which one can interpret as a very sophisticated analogue of a packing tree which is agnostic to  $h$ . These skeletons are also used to derive optimal bounds for the largest subset of a metric space embeddable into  $\ell_2$  with small distortion (known as a non-linear Dvoretzky theorem). [11] asked whether one can improve their bound to  $\delta_h(X) \leq (2 + o(1))\gamma_h(X)$ , where the factor of 2 is tight up to  $o(1)$  factors for an  $n$ -point star-metric with  $h = g$ . Up to the additive constant (which is often  $o(1)$  compared to  $\gamma_h(X)$ ) and the restriction that  $h$  be of log-concave type, Theorem 1.10 resolves their question in the affirmative.

### 1.2.3 Rounding Measures to Trees

Towards proving our main theorem (Theorem 1.6), we show how to round a measure  $\rho$  to chaining and packing trees with values approximately  $\gamma_h(\rho, X)$  and  $\delta_h(\rho, X)$  respectively. As remarked before, the primal rounding strategy was already introduced by Talagrand [21] himself. While the algorithm is simple, it is based on a not terribly intuitive variant of ball partitioning, which is perhaps due to the structure of the object he converts to (an admissible sequence). In the present work, we show that Talagrand’s basic greedy ball partitioning scheme with the functional replaced by a measure, very transparently yields a construction of good chaining trees. The greedy ball partition algorithm iteratively selects centers that maximize the measure of balls of a smaller radius and removes a ball of a larger radius centered at the previously chosen points. One does this until the removed pieces form a partition and then proceeds recursively on those pieces. Here we show that this can be implemented in near linear time in the input size which is  $O(n^2)$  for an  $n$ -point metric space.

► **Theorem 1.11.** *There exists a deterministic algorithm that runs in  $O(n^2 \log n)$  time and given a probability measure  $\rho$  on an  $n$ -point metric space  $X$ , finds a chaining tree  $\mathcal{C}$  such that  $\text{val}_h(\mathcal{C}) \lesssim \gamma_h(\rho, X)$ .*

On the dual side, as far as we are aware there were no rounding strategies to compute packing trees starting from a measure  $\rho$ . The previous approaches for rounding [17, 10, 3] were based on defining a functional that satisfies the previously mentioned “super-chaining” inequality and used a greedy partitioning procedure based on the value of this functional.



This analysis was rather delicate and somewhat mysterious. Moreover, the corresponding functionals in these instances were themselves solutions to optimization problem on the metric space, so implementing this strategy deterministically was rather slow. In fact, Ding, Lee and Peres [3] showed that using a carefully constructed dynamic program, one can implement the above strategy and compute a packing tree in polynomial time in the input size when the metric space is given as the input. Although, they don't specify a precise bound on the running time, one can directly infer a  $O(n^4)$  deterministic running time for building a packing tree; with an additional observation, this can in fact be improved to a  $O(n^3 \log n)$  bound.<sup>2</sup>

In this work, we revisit the above approach and show that in fact, one can round a probability measure  $\rho$  on the metric space to a packing tree with approximately the same value as  $\delta_h(\rho, X)$ . This has certain advantages over a rounding strategy using functionals as it can be implemented in near linear time in the input size. Moreover, this rounding algorithm is quite similar to the primal rounding algorithm and in our opinion clarifies why such a construction works. In particular, the basic strategy of choosing centers that maximize the measure of a smaller ball remains exactly the same – one just selects smaller balls to add as children and recurses on them instead, followed by some post-processing.

► **Theorem 1.12.** *There exists a deterministic algorithm that runs in  $O(n^2 \log n)$  time and given a probability measure  $\rho$  on an  $n$ -point metric space  $X$ , finds a  $(1/10)$ -packing tree  $\mathcal{T}$  such that  $\delta_h(\rho, X) \lesssim \text{val}_h(\mathcal{T})$ .*

### 1.2.4 Proof of the Combinatorial Min-Max Theorem (Theorem 1.6)

Of course, without a way to compute measures which have almost optimal values for  $\gamma_h(\rho, X)$  and  $\delta_h(\rho, X)$ , the above rounding algorithms would not have been very useful. Fortunately, we can obtain almost optimal primal and dual measures  $\mu$  and  $\nu$  by solving the saddle point formulation (1.20) of  $\gamma_h(X)$ . Plugging the measure  $\mu$  in Theorem 1.11 gives us a chaining tree  $\mathcal{C}^*$  such that  $\text{val}_h(\mathcal{C}^*) \lesssim \gamma(\mu, X) \asymp \gamma_h(X)$ .

On the dual side, the measure  $\nu$  is not enough as it might not be a good solution to the approximate dual  $\delta_h(X)$ . However, using Theorem 1.10, one can find a set  $S \subset X$  such that the probability measure  $\nu_S$  obtained by conditioning  $\nu$  on the set  $S$  satisfies  $\delta_h(\nu_S, X) \asymp \gamma_h(X)$ . Plugging the measure  $\nu_S$  in Theorem 1.12, gives us a packing tree  $\mathcal{T}^*$  satisfying  $\delta_h(\nu_S, X) \lesssim \text{val}_h(\mathcal{T}^*)$ . Combining the two yields that

$$\text{val}_h(\mathcal{C}^*) \lesssim \gamma_h(X) \asymp \delta_h(\nu_S, X) \lesssim \text{val}_h(\mathcal{T}^*). \quad (1.26)$$

Recall that the weak duality relation (1.19) between chaining and packing trees implies the reverse inequality for  $\text{val}_h(\mathcal{T}) \lesssim \text{val}_h(\mathcal{C})$  for any chaining tree  $\mathcal{C}$  and any packing tree  $\mathcal{T}$ . Thus, (1.26) gives us the combinatorial min-max statement given by Theorem 1.6. Moreover, this can be made algorithmic by solving the saddle point formulation and using the algorithm to find the set  $S$  given by Theorem 1.10. The details for solving the saddle point formulation are given in the appendix of the full version of this paper and the algorithm to find the set  $S$  is presented in the proof of Theorem 1.10.

<sup>2</sup> Their running time is  $O(n^2)$  times the number of “distance scales” in the metric, meaning the number of dyadic intervals  $[2^k, 2^{k+1})$  containing at least one distance  $d(u, v)$  in the metric. To bound the number of distance scales by  $O(n \log n)$ , compute a minimum spanning tree  $T$  in the complete graph describing the metric. Consider any edge  $e = \{v, w\}$  not in the tree, and let  $m(v, w)$  denote an edge on the path between  $v$  and  $w$  in  $T$  of maximum length. Then  $e$  has length at least the length of  $m(v, w)$  since  $T$  is an MST, but not more than  $n$  times larger. That is, the distance scale of  $e$  is in a range of size  $O(\log n)$  of the distance scale of  $m(v, w)$ . Now consider assigning each non-tree edge  $\{v, w\}$  to  $m(v, w)$ ; there are  $O(\log n)$  distance scales assigned to each tree edge, so  $O(n \log n)$  in total.

### 1.3 Organization

In Section 2, we list some basic notation as well as the main useful properties of chaining functionals of log-concave type. In Section 3, we simplify the dual program  $\gamma_h^*(X)$ , proving the main inequalities relating it to the entropic and simplified duals  $\delta_h^{\text{Ent}}(X)$  and  $\delta_h(X)$  respectively. Other details including near-linear time rounding algorithms from measures to chaining trees and packing trees, deterministic construction of Johnson-Lindenstrauss projections achieving Gordon's bound, using a black-box solver to compute nearly-optimal primal and dual measures for the saddle-point formulation of  $\gamma_h(X)$ , and proofs of other technical statements can be found in the full version of this paper.

## 2 Preliminaries

**Notation.** Throughout this paper,  $\log$  denotes the natural logarithm unless the base is explicitly mentioned. We use  $[k]$  to denote the set  $\{1, 2, \dots, k\}$ . For a vector  $z \in \mathbb{R}^n$ , we will use  $z_i$  or  $z(i)$  interchangeably to denote the  $i$ -th coordinate of  $z$ . Given a probability measure  $\mu$  over a set  $X$ , we use  $\mathbb{E}_{x \sim \mu}[f(x)]$  to denote the expectation of  $f(x)$  where  $x$  is sampled from  $\mu$ .

### 2.1 Properties of Chaining Functionals

Recall that we work with a chaining functional  $h : (0, 1] \rightarrow \mathbb{R}_+$  defined by  $h(p) = F^{-1}(p)$ , where  $F(s) = \int_s^\infty f(x)dx$  and where  $f$  is non-decreasing and continuous probability density on  $\mathbb{R}_+$ . We assume throughout the rest of the paper that  $h$  is of log-concave type, i.e., that  $f$  is log-concave.

The fundamental property of such functionals, that we make extensive use of, is the following:

► **Proposition 2.1.** *For a chaining functional  $h$  of log-concave type,*

$$h(ab) \leq h(a) + h(b) \text{ for } a, b \in (0, 1].$$

Before giving a proof, we note that the above property appears in [10] as the base assumption for the chaining functionals they consider. The above shows that this condition is very natural and applies to a wide variety of functionals.

**Proof.** Let us define  $\varphi : [0, \infty) \rightarrow [0, \infty)$  as  $\varphi(t) := h(e^{-t})$ . We will show that  $\varphi$  is concave on its domain, which implies that  $h$  is sub-additive as

$$h(ab) = \varphi\left(\log \frac{1}{ab}\right) \leq \frac{1}{2}\varphi\left(2\log \frac{1}{a}\right) + \frac{1}{2}\varphi\left(2\log \frac{1}{b}\right) \leq \varphi\left(\log \frac{1}{a}\right) + \varphi\left(\log \frac{1}{b}\right) = h(a) + h(b),$$

where both inequalities follow from concavity and  $\varphi(0) = 0$ .

To see that  $\varphi$  is concave, we show that  $\varphi'(t) = -e^{-t}h'(e^{-t}) = \frac{e^{-t}}{f(F^{-1}(e^{-t}))}$  is a decreasing function of  $t$ . Substituting  $x = F^{-1}(e^{-t})$ , and noting that  $f$  is decreasing, it suffices to show that  $\frac{F(x)}{f(x)}$  is decreasing for  $x$  on the positive real line. Taking arbitrary  $0 \leq x_1 \leq x_2$ , we have that

$$\frac{F(x_1)}{f(x_1)} = \int_0^\infty \frac{f(x_1+t)}{f(x_1)} dt \geq \int_0^\infty \frac{f(x_2+t)}{f(x_2)} dt = \frac{F(x_2)}{f(x_2)},$$

where the inequality follows from the following elementary property of non-negative log-concave functions: for any four points  $a \leq b \leq c \leq d$ , we have that  $f(b)f(c) \geq f(a)f(d)$  which in the above scenario implies that  $\frac{f(x_1+t)}{f(x_1)} \geq \frac{f(x_2+t)}{f(x_2)}$ . This completes the proof of the proposition. ◀

When working on a metric space  $(X, d)$  with a chaining functional  $h$ , we observe the following rescaling symmetry: for any constant  $\beta > 0$ , if we replace  $h$  by  $\beta h$  (equivalently, the density function  $f$  is replaced by  $z \rightarrow \beta f(\beta z)$ ) and  $d(u, v)$  by  $d(u, v)/\beta$  for all  $u, v \in X$ , the values of the various quantities we consider remain unaffected. In particular, if we have an underlying process  $(Z_x)_{x \in X}$  amenable to  $X$  and  $f$  as in (1.10), the condition  $\mathbb{P}[|Z_{x_1} - Z_{x_2}| \geq d(x_1, x_2)s] \leq F(s)$  is identical for both scalings, and the values of the various programs and trees we consider remain unchanged.

So from now on, we make the convenient choice that  $f(0) = 1$ . This implies that the (left) derivative  $h'(1) = -1/f(0) = -1$ . We maintain this normalization for the remainder of the paper.

Given this normalization, the following useful bound is easy to show.

► **Proposition 2.2.** *For every  $a \in (0, 1]$ , we have that*

$$-1 \leq ah'(a) \leq 0 \text{ and } h(a) \leq \log(1/a).$$

**Proof.** As  $h$  is decreasing,  $h'(a) \leq 0$  for  $a \in (0, 1]$ . Therefore, for any  $a$ ,

$$\begin{aligned} ah'(a) &= \lim_{\epsilon \rightarrow 0^+} \frac{a(h(a) - h(a(1 - \epsilon)))}{a - a(1 - \epsilon)} \geq \lim_{\epsilon \rightarrow 0^+} \frac{h(a) - h(a) - h(1 - \epsilon)}{1 - (1 - \epsilon)} \\ &= \lim_{\epsilon \rightarrow 0^+} \frac{h(1) - h(1 - \epsilon)}{1 - (1 - \epsilon)} = h'(1) = -1, \end{aligned} \quad (2.1)$$

where the first inequality follows from sub-multiplicativity and the last equality holds since  $h(1) = 0$ . The first statement in the proposition follows.

To see the second statement, one can observe that as  $h$  is decreasing, (2.1) implies the following differential inequality:  $h'(a) \geq -\frac{1}{a}$  for every  $a \in (0, 1]$ . Together with the boundary condition that  $h(1) = 0$ , this implies that  $h(a) \leq \log(1/a)$ . ◀

### 3 Dual simplifications

In this section, we provide the main arguments that allow for rounding a solution to  $\gamma_h^*(X)$  to a solution to the simplified dual, via the entropic dual, as described in Section 1.2.2.

**Proof of Lemma 1.7.** Our goal is to prove that taking  $\mu = \nu$  in (1.22) does not cause too much error. We will prove this on a “per scale” basis:

► **Lemma 3.1.** *For any probability measures  $\mu$  and  $\nu$  on  $X$ , and any  $r > 0$ , we have*

$$\int_X h(\nu(B(x, 2r))d\nu(x) \leq \int_X h(\mu(B(x, r)))d\nu(x) + 1/e.$$

Lemma 1.7 follows easily from this:

$$\begin{aligned} \int_X \int_0^\infty h(\nu(B(x, r)))drd\nu(x) &\leq \min_\mu \int_X \int_0^{\mathbf{D}(X)} (h(\mu(B(x, r/2))) + 1/e)drd\nu(x) \\ &\leq 2 \min_\mu \int_X \int_0^\infty h(\mu(B(x, r)))drd\nu(x) + \mathbf{D}(X)/e. \end{aligned}$$

**Proof of Lemma 3.1.** We first recast the statement in graph theoretic terms. Let us define a bipartite graph on the vertex set  $X_1 \cup X_2$  where each  $X_i$  for  $i \in [2]$  is a copy of the index set  $X$ . We add an edge between two vertices  $x_1 \in X_1, x_2 \in X_2$  if  $d(x_1, x_2) \leq r$  – note that

every vertex has an edge incident to it. Define the weight of a vertex  $x \in X_i$  as  $\mu(x)$  and the weight of a vertex  $x \in X_2$  as  $\nu(x)$ . Let  $E$  denote the set of edges in the graph, let  $N(S)$  be the neighbors of a subset of the vertices  $S \subset X_1 \cup X_2$  and let  $N^2(S) = N(N(S))$ . For brevity, we will write  $N(x)$  instead of  $N(\{x\})$  for singleton sets.

With this setup, the statement we want to prove is

$$\sum_{x \in X_2} \nu(x) h(\nu(N^2(x))) \leq \sum_{x \in X_2} \nu(x) h(\mu(N(x))) + 1/e. \quad (3.1)$$

To prove the above, we will use the following structural result which is a consequence of the theory of principal sequences of matroids [6], applied to transversal matroids; we include a self-contained proof in the full version of the paper.

► **Proposition 3.2.** *There exist sequences  $\emptyset = S_0 \subset S_1 \subset \dots \subset S_k = X_1$  and  $0 < \beta_1 < \beta_2 < \dots < \beta_k$ , such that*

1.  $\beta_i \mu(S_i \setminus S_{i-1}) = \nu(N(S_i) \setminus N(S_{i-1}))$  for all  $i \in [k]$ .
2. For all  $i \in [k]$  and  $A \subseteq X_1 \setminus S_{i-1}$ , we have that  $\beta_i \mu(A) \leq \nu(N(A) \setminus N(S_{i-1}))$ .

The proposition above can be viewed as a kind of strengthening of Hall's theorem. For instance, if there is a fractional perfect matching between  $\mu$  and  $\nu$ , i.e., a way of transporting mass distributed according to  $\mu$  on  $X_1$  along edges of the graph to yield precisely the distribution  $\nu$  on  $X_2$ , then the claim will be satisfied with  $k = 1$ ,  $S_1 = X_1$  and  $\beta_1 = 1$ , for in this case,  $\mu(A) \leq \nu(N(A))$  for any  $A \subseteq X_1$  (essentially the easy direction of Hall's theorem). If there is no fractional perfect matching, Hall's theorem implies the existence of a set  $S \subset X_1$  with  $\nu(N(S)) < \mu(S)$ . In the proposition,  $S_1$  is the “least matchable” set: only a  $\beta_1$  fraction of the mass in  $S_1$  can be transported to  $N(S_1)$ . The full sequence is then obtained by removing  $S_1$  and  $N(S_1)$  and repeating on the remainder.

Taking the sequences guaranteed by the proposition, define  $\tilde{\beta}_i := \min\{\beta_i, 1\}$ .

We split the left hand side of (3.1) as follows:

$$\sum_{x \in X_2} \nu(x) h(\nu(N^2(x))) = \sum_{i=1}^k \sum_{x \in N(S_i) \setminus N(S_{i-1})} \nu(x) h(\nu(N^2(x))).$$

Note that for any  $i \in [k]$ , if  $x \in X_2 \setminus N(S_{i-1})$ , then  $N(x) \subseteq X_1 \setminus S_{i-1}$ , and hence, Proposition 3.2 implies that  $\tilde{\beta}_i \mu(N(x)) \leq \nu(N^2(x))$ . Furthermore, using sub-multiplicativity of  $h$  and Proposition 2.2, we find that for any  $i \in [k]$  and  $x \in N(S_i) \setminus N(S_{i-1})$ ,

$$\begin{aligned} h(\nu(N^2(x))) &= h\left(\mu(N(x)) \cdot \frac{\nu(N^2(x))}{\mu(N(x))}\right) \leq h(\mu(N(x))) + h\left(\min\left\{1, \frac{\nu(N^2(x))}{\mu(N(x))}\right\}\right) \\ &\leq h(\mu(N(x))) + h(\tilde{\beta}_i) \leq h(\mu(N(x))) + \log\left(\frac{1}{\tilde{\beta}_i}\right). \end{aligned}$$

For the first inequality above, sub-multiplicativity is used when  $\frac{\nu(N^2(x))}{\mu(N(x))} > 1$ ; otherwise, the inequality follows because  $h$  is decreasing and  $h(1) = 0$ . (The first inequality does still hold if  $\mu(N(x)) = 0$ , taking the minimum in the second term to have value 1 in this case, again because  $h$  is decreasing.)

The second inequality again uses that  $h$  is decreasing, as well as that  $\tilde{\beta}_i \leq 1$  for every  $i \in [k]$ ; and the final inequality uses Proposition 2.2.

### 73:18 Majorizing Measures for the Optimizer

Let  $\ell$  be maximal such that  $\tilde{\beta}_\ell < 1$ ; so  $\beta_i = \tilde{\beta}_i$  for  $i \leq \ell$ . Summing the last inequality over all  $i \in [k]$  and  $x \in N(S_i) \setminus N(S_{i-1})$ , we obtain

$$\begin{aligned} \sum_{x \in X_2} \nu(x) h(\nu(N^2(x))) &\leq \sum_{x \in X_2} \nu(x) h(\mu(N(x))) + \sum_{i=1}^k \nu(N(S_i) \setminus N(S_{i-1})) \cdot \log \left( \frac{1}{\tilde{\beta}_i} \right) \\ &= \sum_{x \in X_2} \nu(x) h(\mu(N(x))) + \sum_{i=1}^{\ell} \tilde{\beta}_i \mu(S_i \setminus S_{i-1}) \cdot \log \left( \frac{1}{\tilde{\beta}_i} \right) \\ &\leq \sum_{x \in X_2} \nu(x) h(\mu(N(x))) + \left( \max_{\beta \in (0,1]} \beta \log \left( \frac{1}{\beta} \right) \right) \sum_{i=1}^{\ell} \mu(S_i \setminus S_{i-1}), \end{aligned}$$

where the second line follows from Proposition 3.2. From this (3.1) readily follows as  $\sum_{i=1}^{\ell} \mu(S_i \setminus S_{i-1}) \leq \sum_{i=1}^k \mu(S_i \setminus S_{i-1}) = \mu(X_1) = 1$  and  $\max_{\beta \in (0,1]} \beta \log \left( \frac{1}{\beta} \right) = 1/e$ .  $\blacktriangleleft$

**Proof of Lemma 1.9.** For convenience, define

$$H(\mu, t) := \int_0^\infty h(\mu(B(t, r))) dr \quad \text{and} \quad H(\mu, \nu) := \int_X H(\mu, t) d\nu(t).$$

Start by setting  $S = \{x \in X : \nu(x) > 0\}$ . Consider the following greedy algorithm:

As long as  $H(\nu_S, \nu_S) > \min_{x \in S} H(\nu_S, x) + \mathbf{D}(X)$ , choose  $s \in S$  so that  $H(\nu_S, s)$  is minimized, and remove  $s$  from  $S$ .

Note that  $H(\nu_S, \nu_S) \leq \min_{x \in S} H(\nu_S, x) + \mathbf{D}(X)$  when this terminates, since it is vacuous for  $S = \emptyset$ .

We will now show that  $H(\nu_S, \nu_S)$  can only increase during the progression of the algorithm. This suffices to prove the lemma, since then upon termination

$$H(\nu, \nu) \leq H(\nu_S, \nu_S) \leq \min_{x \in S} H(\nu_S, x) + \mathbf{D}(X).$$

So, consider a moment in the algorithm where  $s \in S$  is about to be removed from  $S$ , yielding  $S' := S \setminus \{s\}$ . From our choice of  $s$ ,

$$H(\nu_S, \nu_S) > H(\nu_S, s) + \mathbf{D}(X). \quad (3.2)$$

Let  $\alpha = 1/(1 - \nu_S(s))$ ; so  $\nu_{S'}(t) = \alpha \nu_S(t)$  for  $t \neq s$ , and  $\nu_{S'}(s) = 0$ . Thus

$\triangleright$  **Claim 3.3.** For every  $t \in X$ ,

$$H(\nu_{S'}, t) \geq H(\nu_S, t) - \mathbf{D}(X) \cdot (\alpha - 1).$$

**Proof.** Recall that  $h$  is convex and satisfies  $-1 \leq ah'(a) \leq 0$  for any  $a \in (0, 1]$  by Proposition 2.2. Thus for any  $z \in (0, 1]$ ,

$$\begin{aligned} h(z) &\geq h(z/\alpha) + z(1 - 1/\alpha)h'(z/\alpha) \\ &= h(z/\alpha) - (\alpha - 1) \left| (z/\alpha)h'(z/\alpha) \right| \geq h(z/\alpha) - (\alpha - 1). \end{aligned} \quad (3.3)$$

Now notice that for any  $T \subseteq S$ , we have that  $\nu_S(T) \geq \nu_{S'}(T)/\alpha$ . Therefore, since  $h$  is decreasing,

$$\begin{aligned} H(\nu_S, t) &= \int_0^{\mathbf{D}(X)} h(\nu_S(B(t, r))) dr \\ &\leq \int_0^{\mathbf{D}(X)} h(\nu_{S'}(B(t, r))/\alpha) dr \\ &\leq \int_0^{\mathbf{D}(X)} h(\nu_{S'}(B(t, r))) dr + \mathbf{D}(X) \cdot (\alpha - 1) = H(\nu_{S'}, t) + \mathbf{D}(X) \cdot (\alpha - 1), \end{aligned}$$

where the last inequality follows from (3.3).  $\triangleleft$

We can now compare  $H(\nu_{S'}, \nu_{S'})$  with  $H(\nu_S, \nu_S)$ :

$$\begin{aligned} H(\nu_{S'}, \nu_{S'}) &\geq H(\nu_S, \nu_{S'}) - \mathbf{D}(X) \cdot (\alpha - 1) && \text{(by Claim 3.3)} \\ &= \frac{1}{1-\nu_S(s)} (H(\nu_S, \nu_S) - \nu_S(s)H(\nu_S, s)) - \frac{\nu_S(s)}{1-\nu_S(s)} \cdot \mathbf{D}(X) \\ &\geq \frac{1}{1-\nu_S(s)} (H(\nu_S, \nu_S) - \nu_S(s)(H(\nu_S, \nu_S) - \mathbf{D}(X))) - \frac{\nu_S(s)}{1-\nu_S(s)} \cdot \mathbf{D}(X) && \text{(by (3.2))} \\ &= H(\nu_S, \nu_S), \end{aligned}$$

which finishes the proof of Lemma 1.9.

**Runtime Analysis.** It is easily seen that the algorithm can be implemented in  $O(n^3)$  time. First, by pre-processing the input, we may assume that the pairs of points are sorted by their pairwise distances – this only adds an additional overhead of  $O(n^2 \log n)$ . Then, as in each iteration we remove one element, there are  $O(n)$  iterations to compute the final set  $S'$ . Furthermore, in each of these iterations, we are required to compute a minimizer  $s$  of  $H(\nu_S, x)$ . This takes  $O(n^2)$  time since for each  $x \in S$ , one can compute the measure of all possible balls  $B(x, r)$  in  $O(n)$  time using a straightforward dynamic program.

---

## References

- 1 Witold Bednorz. A theorem on majorizing measures. *The Annals of Probability*, 34(5):1771–1781, 2006.
- 2 Witold Bednorz. The majorizing measure approach to sample boundedness. *Colloquium Mathematicum*, 139, November 2012.
- 3 Jian Ding, James R Lee, and Yuval Peres. Cover times, blanket times, and majorizing measures. *Annals of mathematics*, 175(3):1409–1471, 2012.
- 4 Richard M Dudley. The sizes of compact subsets of Hilbert space and continuity of Gaussian processes. *Journal of Functional Analysis*, 1(3):290–330, 1967.
- 5 Xavier Fernique. Régularité des trajectoires des fonctions aléatoires gaussiennes. In *Ecole d’Eté de Probabilités de Saint-Flour IV–1974*, pages 1–96. Springer, 1975.
- 6 Satoru Fujishige. *Theory of Principal Partitions Revisited*, pages 127–162. Springer Berlin Heidelberg, 2009.
- 7 Yehoram Gordon. On Milman’s inequality and random subspaces which escape through a mesh in  $\mathbb{R}^n$ . In *Geometric aspects of functional analysis*, pages 84–106. Springer, 1988.
- 8 Olivier Guédon and Artem Zvavitch. *Supremum of a Process in Terms of Trees*, pages 136–147. Springer Berlin Heidelberg, 2003.

- 9 Haotian Jiang, Yin Tat Lee, Zhao Song, and Sam Chiu-wai Wong. An improved cutting plane method for convex optimization, convex-concave games, and its applications. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020*, pages 944–953. ACM, 2020.
- 10 Michel Ledoux and Michel Talagrand. *Probability in Banach Spaces: Isoperimetry and Processes*, volume 23. Springer Science & Business Media, 1991.
- 11 Manor Mendel and Assaf Naor. Ultrametric subsets with large Hausdorff dimension. *Inventiones mathematicae*, 192, June 2011.
- 12 Manor Mendel and Assaf Naor. Ultrametric skeletons. *Proceedings of the National Academy of Sciences*, 110(48):19256–19262, 2013.
- 13 Vitali D Milman. A new proof of A. Dvoretzky’s theorem on cross-sections of convex bodies. *Funkcional. Anal. i Prilozhen*, 5:28–37, 1971.
- 14 Samet Oymak, Benjamin Recht, and Mahdi Soltanolkotabi. Isometric sketching of any set via the restricted isometry property. *Information and Inference: A Journal of the IMA*, 7(4):707–726, 2018.
- 15 David Slepian. The one-sided barrier problem for Gaussian noise. *Bell System Technical Journal*, 41(2):463–501, 1962.
- 16 Vladimir Nikolaevich Sudakov. Gaussian random processes and measures of solid angles in Hilbert space. In *Doklady Akademii Nauk*, volume 197, pages 43–45. Russian Academy of Sciences, 1971.
- 17 Michel Talagrand. Regularity of Gaussian processes. *Acta mathematica*, 159:99–149, 1987.
- 18 Michel Talagrand. Sample boundedness of stochastic processes under increment conditions. *The Annals of Probability*, pages 1–49, 1990.
- 19 Michel Talagrand. A simple proof of the majorizing measure theorem. *Geometric & Functional Analysis GAFA*, 2(1):118–125, 1992.
- 20 Michel Talagrand. Majorizing measures: the generic chaining. *Ann. Probab.*, 24(3):1049–1103, July 1996.
- 21 Michel Talagrand. Majorizing measures without measures. *Ann. Probab.*, 29(1):411–417, February 2001.
- 22 Ramon van Handel. Probability in High Dimensions. Lecture Notes. Princeton University, 2016. URL: <https://web.math.princeton.edu/~rvan/APC550.pdf>.



# Randomness and Fairness in Two-Sided Matching with Limited Interviews

Hedyeh Beyhaghi

Toyota Technological Institute at Chicago, IL, USA  
hedyeh@ttic.edu

Éva Tardos

Department of Computer Science, Cornell University, Ithaca, NY, USA  
eva.tardos@cornell.edu

---

## Abstract

We study the outcome in a matching market where both sides have limited ability to consider options. For example, in the national residency matching program, doctors are limited to apply to a small set of hospitals, and hospitals are limited by the time required to interview candidates.

Our main findings are the following: (1) In markets where jobs can only consider a limited number of candidates for interview, it increases the size of the resulting matching if the system has a limit on the number of applications a candidate can send. (2) The fair system of all applicants being allowed to apply to the exact same number of positions maximizes the expected size of the matching. More particularly, starting from an integer  $k$  as the number of applications, the matching size decreases as a few applicants are allowed to apply to one additional position (and then increases again as they are all allowed to apply to  $k + 1$ ). Although it seems natural to expect that the size of the matching would be a monotone increasing and concave function in the number of applications, our results show that neither is true. These results hold even in a market where a-priori all jobs and all candidates are equally likely to be good, and the judgments of different employers and candidates are independent.

Our main technical contribution is computing the expected size of the matching found via the deferred acceptance algorithm as a function of the number of interviews and applications in a market where preferences are uniform and independent. Through simulations we confirm that these findings extend to markets where rankings become correlated after the interviews.

**2012 ACM Subject Classification** Theory of computation → Algorithmic mechanism design

**Keywords and phrases** Matching with Short Lists, Stable Matching, Balls in Bins Problem

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.74

**Funding** *Hedyeh Beyhaghi*: The author was supported in part by NSF grant CCF-1563714.

*Éva Tardos*: Supported in part by NSF grants CCF-1408673, CCF-1563714 and AFOSR grant FA9550-19-1-0183.

**Acknowledgements** The authors are thankful to Mark Braverman and Robert Kleinberg for fruitful discussions.

## 1 Introduction

Matching is a fundamental paradigm in a variety of real-world situations and arises in various domains, e.g., students applying to attend schools, or colleges, applicants get matched with jobs at various job markets, and medical residents get matched with hospitals, just to name a few. In some of these domains, matchings are found through a centralized algorithm such as the deferred acceptance algorithm by Gale and Shapley [5]. Prominent examples include: national residency matching program (NRMP) for assigning medical students to hospital residency programs; schools assignments in some cities the US; and college admissions in



© Hedyeh Beyhaghi and Éva Tardos;  
licensed under Creative Commons License CC-BY  
12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 74; pp. 74:1–74:18



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

some countries [10]. Although the deferred acceptance algorithm is designed for centralized markets, it can also be thought of as a rough model of how uncoordinated matching markets as applicants collect offers and as they turn down offers, new offers may be made.

An important aspect that is not modeled under the classical deferred acceptance algorithm is the fact that both sides of the market can only consider a limited set of options. This is the issue we will focus on in the current paper. The deferred acceptance algorithm is typically studied assuming that participants express their full list of ordered preferences. However, typically both sides of the match are limited in the number of options they can consider. In some markets, such as the national residency matching program, there are application limits imposed by the system. But even without such explicit limits, preparing applications and interviewing applicants is costly and time consuming; and hence typically extremely limited. Applications, e.g. for college admission, often require writing specialized essays. Residency programs, as well as some of the colleges, and almost all jobs, interview their applicants, which requires significant time and effort. In large matching markets it is not feasible for either side to consider a large set. For instance, in NRMP, there are nearly 5000 residency programs and applicants have to limit their choices. Similarly, interviewing consumes significant time of the hospitals, and so hospitals can only grant interviews to a limited set of doctors. An alternative view of the limitations of the employer side is a limit on the offers they can make for one position: based on their interviews with all or a subset of the applicants, they may consider only a limited number of their favorite applicants. Such selective offer making is observed in the academic job markets, e.g., when some high ranked departments prefer to wait till next year, rather than make offers to applicants they liked less.

The goal of this paper is to study the effect of the number of applications and the number of interviews on the resulting matching. We think of our procedure as a simplified version of the matching process of the national residency matching program. In this program, the matching mechanism has two stages. In the first stage, doctors express their interest to a set of hospitals, and hospitals choose to interview a subset of the applicants. In the second stage, both doctors and hospitals submit their ordered preference list over the set they interviewed with. The centralized matching market uses these preference lists to perform the deferred acceptance algorithm and outputs the final matching.

We will primarily consider a *symmetric* market, where doctors and hospitals are similar in terms of popularity, and as a primary metric of the social welfare of the system, we will mainly consider the size of the matching found. In asymmetric markets, where some applicants and some jobs are a-priori better and preferred by most applicants, the limit on the number of choices gives rise to strategic issues resulting in top candidates getting too many offers, and a number of jobs remaining unfilled – see the discussion in related literature section. In this paper we isolate the effect of limitation from strategicness. We focus on large symmetric markets, where it will be equilibrium for both sides to report their true preferences. We find a number of surprising effects of this limited choice, resulting in two policy recommendations.

- With limited ability granting interviews, social welfare may be maximized by also severely limiting the number of applications one applicant can send.
- The best social welfare is achieved by a fair system, where all applicants can send the same number of applications. Allowing a small subset to send even just one additional application decreases the overall welfare of the system.

## 1.1 Main Results

The main contribution of the paper is studying the effect of limitation on the number of applications and interviews in matching markets. In contrast to the majority of the previous work that focused on correlated preferences, we completely abstract away the role of correlation, consider a purely random model, and study the role of randomness extensively. The main results of the paper are as following.

- We compute the expected size of matching resulted from the deferred acceptance algorithm as a function of the limit on the number of applications and interviews. See Proposition 6, Theorem 10, and Proposition 20.
- We find the optimal way for distributing the applications among applicants, when the total number of allowed applications is fixed. The maximum size of matching is achieved when all doctors apply to the same number of positions or two consecutive integers. See Proposition 11.
- With a fixed number of interviews per position, the size of the matching as a function of the number of applications is a scallop-shape figure (see Figures 2 to 5), where the function between consecutive integers is U-shaped.

### Contribution to Balls in Bins Literature

Numerous findings in the matching literature utilize the results and techniques from *balls in bins* problems. Although this paper is written in the matching language, it has analogues in balls in bins language and contributes to that literature as well. For example consider the following problem: There is a set of  $n$  balls and  $n$  bins. Each ball  $d$  has  $k_d$  copies. Each bin has capacity one. All copies of the balls are thrown independently and uniformly at random to the bins. Each bin only accepts one ball at random among those that have been thrown to it. We are interested in  $S$  which is the expected number of unique balls (counting only a single copy from each ball) that land in bins. Various questions can be asked in this scenario. For example, what is the optimal value of  $S$ ? What distribution for  $k_d$  achieves this value? Section 3, which is a warm-up for our main result, completely analyses this problem and surprisingly shows the optimal value of  $S$  is  $\approx 0.68$  for large enough  $n$  and it is achieved when there are 3 copies from each ball,  $k_d = 3$ .

## 1.2 Related Work

Papers studying matching with short lists have different viewpoints on how the short lists are selected. For example, Immorlica and Mahdian [6], Kojima and Pathak [8], and Arnosti [1] study matchings when preference lists are inherently short, i.e. the participants prefer to stay unmatched rather than being matched outside their preference lists. In contrast, Avery and Levin [2], Kadam [7], Drummond et al. [4], and Beyhaghi et al. [3] study how applicants make the strategic choice to select a limited number of positions for their short preference lists. In this paper, because of the uniform preference of participants we do not focus on their strategic behavior.

Arnosti [1], Lee and Schwarz [9], and Kadam [7] study efficiency of matching either as matching size or social welfare in presence of short lists. Arnosti evaluates social welfare under different preference models. However, unlike our paper, in [1] the preference lists are limited only for one side of the market. Similar to us, Lee and Schwarz study matching size in a setting with an interview stage, where the ex-ante preferences are i.i.d. However, they solve the problem of some doctors not receiving any offer while others receive many, by coordinating the set of doctors that each hospital interviews. In contrast, in this paper, we

assume that the interviews are selected in a decentralized way without imposing coordination; and our solution is to limit the number of applications and let the applicants have the same number of applications. Similar to our work, Kadam studies a model with limit on both sides of the market and shows that limiting the length of lists can have positive effects of the size of matching. However, in [7] these effects are due to almost common preferences: When some doctors are more preferred, with a stricter limit on the interviews for doctors, the more preferred doctors do not accept interviews with less preferred hospitals. In this paper, we show that limiting the length can be helpful in the exactly opposite case where all participants are identical in terms of popularity.

## Roadmap

The rest of the paper is organized as follows. Section 2 discusses the model and preliminaries. In Section 3, as a warm-up we analytically compute the size of the matching when the number of interviews is limited to one. In Section 4, we compute the size of the matching for arbitrary number of interviews. Section 5 discusses two extensions to the main result in Section 4. In Section 5.1, we show that the same phenomena remain true in unbalanced markets (when the two sides are of different size). In Section 5.2, we study a model where the hospitals and doctors preferences are a-priori uniform, but become correlated after interviewing.

## 2 Preliminaries

We explain our model in Section 2.1, and discuss the unlimited interview case in Section 2.2.

### 2.1 Model

There is a finite set of doctors  $\mathcal{D}$  and a finite set of hospitals  $\mathcal{H}$ , with  $|\mathcal{D}| = n$  and  $|\mathcal{H}| = rn$ . We are interested in one-to-one matchings between doctors and hospitals.

#### Description of the Game

We consider a two-stage matching mechanism. In the first stage, each doctor applies to a set of hospitals and requests interviews. Hospitals then choose a subset from their applications to conduct interviews. In the interview process, doctors and hospitals refine their preferences. In the second stage, at the end of interviews, both doctors and hospitals order the list that they interviewed with, based on their preferences. They submit their ordered preferences to the system which performs a doctor-proposing deferred acceptance algorithm to determine the final assignment of doctors to hospitals. The algorithm starts with all doctors unmatched. In each step, the algorithm simulates all unmatched doctors, who have not yet exhausted their options, propose to their most preferred hospital among those to which he/she has not yet proposed. Now the algorithm simulates that each hospital tentatively accepts their most preferred doctor from the doctors now proposing and the one who has been tentatively assigned to this hospital, and rejects all other doctors. The procedure is repeated until all unmatched doctors have been rejected from every hospital in their lists.

We use the following terminology throughout the whole paper.

**Application:** The procedure in the first stage where the doctors submit an application and ask for interview.

**Grant interview or reject for interview:** The procedure in the first stage where the hospitals grant interviews to a number of doctors that have applied for the position and reject the rest.

**Valid application:** Any application that is granted an interview is a valid application. Other applications are *invalid*.

**Proposal:** The procedure in the second stage that simulates doctors proposing to hospitals as steps of the deferred acceptance algorithm.

**Offer:** In the special case where the hospitals limit their number of interviews to one, conducting interviews is of no use because the hospital offers the position to the selected doctor in any case. Therefore, we use offer instead of interview in this case.

### Limits on Applications and Interviews

Each doctor  $d$  is allowed to apply to  $k_d$  positions. In the simpler case, there is a universal limit on the number of applications,  $k_d = k$ , inspired by NRMP granting all doctors 10 initial applications. There is a universal limit  $k'$  on the number of interviews each hospital can conduct.

### Random Preferences

For the most of the paper, we assume that everybody's preference comes independently from uniform distribution over the other side. Both doctors and hospitals refine their preference order for the list they interviewed. However, the overall distribution of doctors preferences and hospitals preferences stays uniform and independent. For analytic purposes we can assume that preferences stay unchanged after the interviews.

► **Observation 1.** *Suppose prior to interviews preferences are drawn independently from uniform distributions, and after interviews, the overall distribution of preferences are uniform and independent. In this case, assuming that the preferences remained unchanged leads to the same analytic results including the same matching size.*

► **Observation 2.** *Being truthful is a Bayes Nash equilibrium.<sup>1</sup> We assume both doctors and hospitals follow this strategy and are truthful during the matching procedure: In the first stage, doctors apply to their favorite hospitals; and hospitals grant interviews to their favorite applicants. In the second stage, both doctors and hospitals submit the list that they interviewed, ordered from the most preferred to the least preferred.*

### Efficiency Measure

We focus on the size of the matching as our notion of efficiency. We define social welfare as the ratio of size of the matching outcome to the size of the maximum matching. Since we assume all doctors/hospitals prefer to be matched to any hospital/doctor rather than being unmatched, in a maximum-size matching everybody on the less populated side of the market is matched.

► **Definition 3 (Social Welfare).** *We define social welfare of the matching outcome as the ratio of the size of the matching outcome compared to the size of the maximum-size matching. In a two sided market, maximum-size matching will be the size of the smaller side of the market.*

---

<sup>1</sup> For more discussion, see the appendix.

## Large Markets

The results consider outcome in “large markets” as introduced in [1]. Formally, a sequence of markets indexed by  $n$  are considered. In this sequence, the number of applications  $k$ , the number of interviews  $k'$ , and the ratio of number of hospitals to doctors  $r$  are held constant. The  $n^{\text{th}}$  market is characterized by a set of doctors  $\mathcal{D}^n$  and a set of hospitals  $\mathcal{H}^n$ . Consider  $n = |\mathcal{D}^n| = |\mathcal{H}^n|/r$ . The results in this paper study the properties of the matching for  $n \rightarrow \infty$ .

## 2.2 No Limit for Granting Interviews

In this section, we recall an analysis from [3] with  $n$  doctors and  $n$  hospitals in the market. In this case there is no limit on the number of interviews by hospitals and doctors are allowed to list only up to  $k$  hospitals, for  $k \ll n$ . When doctors apply to  $k$  hospitals, Proposition 4 finds the probability  $p$  of a single random proposal resulting to a permanent match as  $n$  grows to infinity. The main idea is that in the limit as  $n$  goes to infinity, the probability of a proposal resulting in a match is independent of the previous proposals of the applicant being rejected.

► **Proposition 4** ([1]). *When each doctor applies to  $k$  hospitals, the probability  $p$  of a single random proposal resulting in a permanent match in the deferred acceptance procedure, satisfies the following equation.*

$$(1 - p)^k = e^{-(1-(1-p)^k)/p} \quad (1)$$

**Proof sketch.** Since the outcome of the deferred acceptance algorithm does not depend on the order in which doctors propose, we may hold out a single doctor  $d$  and run the deferred acceptance algorithm on the remainder of the market. Now consider doctor  $d$  proposing to her favorite position. Her first few proposals may get rejected. Once a hospital accepts her proposal, it may reject a different doctor, who may propose for her next position, etc. We call the resulting sequence of rejections a *rejection chain*. The probability that a proposal of doctor  $d$  causes a rejection chain that gets doctor  $d$  rejected from the hospital that first accepted her vanishes as the market grows, therefore we may assume that  $d$ 's first tentatively accepted proposal will lead to a permanent match. Also, in a large market, the rejection of  $d$ 's first  $m$  proposals does not affect the probability of acceptance of other proposals. Thus, from  $d$ 's perspective, each hospital that she applied to in the first stage, should be available to her with some probability  $p$ , and their availability should be independent. With this argument, the probability that  $d$  matches is  $1 - (1 - p)^k$ , and the expected number of hospitals  $d$  proposes to is

$$1 + (1 - p) + (1 - p)^2 + \dots + (1 - p)^{k-1} = \frac{1}{p}(1 - (1 - p)^k).$$

From the point of view of each hospital, each of these proposals is sent to them roughly with probability  $1/n$ ; thus the probability that a hospital receives at least one proposal is

$$1 - (1 - 1/n)^{\frac{1}{p}(1-(1-p)^k)};$$

which approaches  $1 - e^{-(1-(1-p)^k)/p}$  when  $n \rightarrow \infty$ . Since doctors match with probability  $1 - (1 - p)^k$ , and the number of doctors and hospitals that match must be equal, we have that  $(1 - p)^k = e^{-(1-(1-p)^k)/p}$ . ◀

► **Proposition 5.** *Suppose that either all the doctors apply to the same number of hospitals or a fraction of them apply to  $k$  while others apply to  $k + 1$  hospitals. In a setting where there is no limit on the number of interviews by hospitals and the preferences are uniformly random, the social welfare of the matching is increasing in the expected number of applications.*

**Proof.** See the appendix. ◀

### 3 Warm-Up: Single Offer

In this section, we find the efficiency of matching as a function of the expected number of applications by doctors, when hospitals only select one applicant in the first stage of the mechanism. We begin with the case where all doctors apply to the same number of hospitals,  $k_d = k$ , and then we move to the case where doctors send different number of applications. We find allocating the number of applications equally results in the most efficient matching and allowing a small subset to apply to an extra position or restricting a small set to apply to one less position, results in a smaller matching.

In this part we assume that hospitals only select one of their applicants in the first stage of the matching. As discussed in Observation 2, they choose their favorite applicant. Since interviews are conducted to compare the selected doctors, in this case that only one doctor is selected, interviews are of no use from the hospitals' point of view. Therefore the hospital immediately offers the position to the selected doctor. Consequently, the outcome of the deferred acceptance algorithm can be easily found in this case; each doctor will be matched to his/her favorite hospital among those who gave an offer. Therefore without going through the complication of the deferred acceptance algorithm, each doctor accepts the most preferred offer. This discussion implies that the size of matching in this case, is the number of doctors who receive an offer.

#### 3.1 Same Number of Applications

First, we study the social welfare of the matching with respect to the number of applications by doctors, where all doctors apply to the same number of hospitals,  $k$ .

► **Proposition 6.** *Suppose for all  $d$ ,  $k_d = k$  and  $k' = 1$ . The social welfare equals  $1 - (1 - \frac{(1-e^{-k})}{k})^k$ .*

To illustrate some of the main points in the proof of the proposition we start with a simple example in which doctors are allowed to send one application.

► **Example 7.** When  $k = k' = 1$ , the social welfare of the matching approaches  $(1 - 1/e) \approx 0.63$  in a large market. In a matching between hospitals and doctors the size of matching equals the number of matched hospitals. Each hospital that receives an application is matched. The reason is when a hospital accepts a doctor, that doctor does not have any other offers and accepts the match. The probability of a hospital receiving at least one application is  $1 - (1 - 1/n)^n$  which tends to  $1 - 1/e$  with  $n$  approaching  $\infty$ .

Now, we consider the general case of arbitrary  $k$ . The following definition is central to our proof.

► **Definition 8** (covered hospital). *A hospital is covered if it receives at least one application. The number of covered hospitals equals the number of offers.*



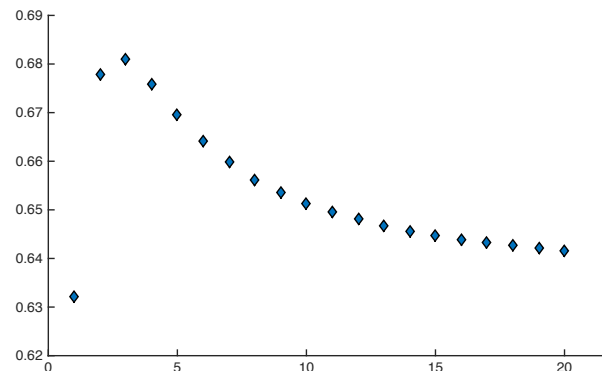
**Proof of Proposition 6.** First, we find the expected number of covered hospitals. By Observation 2, doctors apply to their favorite positions. Since doctors preferences are independent, the set of hospitals that a doctor applies to is independent from other doctors. Although a doctor applies to  $k$  different hospitals and these applications are not technically independent, as  $n$  grows large, the dependence between different applications of a doctor becomes negligible tends to 0. In other words, the probability that a fresh random choice of a doctor is identical to a previous choice approaches to 0. Therefore, we assume that different applications of a doctor are independent. The independence among applications of the same doctor and the set of applications of different doctors, implies the  $nk$  hospitals chosen by applications are selected uniformly and independently at random. The probability of a hospital receiving any application is  $\approx 1 - (1 - 1/n)^{nk}$ , which tends to  $(1 - e^{-k})$  in the limit.

Now, we formulate the probability of a doctor being matched. A doctor is matched if one of its applications turns to an offer. The probability of application  $(d, h)$  turning into an offer depends on the number of applications that hospital  $h$  has received. (Each hospital gives offer to one of its applications.) However, doctors have no knowledge about the number applications a hospital receives and to them all hospitals look identical in terms of the number of applications they have. Thus, from a doctor's perspective, each of their applications has the same probability of turning into an offer. Let  $p$  be the probability of an application turning to an offer. The probability of a doctor being matched equals  $1 - (1 - p)^k$ .

So far, we found the probability of a hospital being covered and the probability of a doctor being matched as a function of  $p$ . Now, we can relate these two terms. The main observation here is that the probability of an application turning to an offer,  $p$ , equals the ratio between the number of offers and all applications. Therefore,  $p = \frac{1-e^{-k}}{k}$ .

The expected social welfare of matching is equal to the probability of a doctor being matched, which is  $(1 - (1 - \frac{1-e^{-k}}{k})^k)$ . ◀

Figure 1 shows the size of matching with respect to the number of applications.



■ **Figure 1** Size of matching with respect to the number of applications, when hospitals make only one offer.

### Positive and negative effects of having more applications

As shown in Figure 1, having more applications can have both positive and negative effects: As a positive effect, with more applications the number of hospitals who receive an application increases. This can potentially increase the size of matching. On the other hand, as the number of applications increases, the hospitals become more congested. Since the total number

of offers is limited by the number of hospitals (each hospital makes at most one offer), this causes an increase in the probability of rejection of an application. This may increase the number of doctors with no offer, leading to a negative effect on the size of matching. As seen in the picture, The biggest jump in the size of matching occurs when moving from one application to two applications. When increasing the number of applications from one to two, the fraction of covered hospitals changes from  $1 - 1/e \approx 0.63$  to  $1 - 1/e^2 \approx 0.86$ , which is the highest increase when adding more applications. This causes the highest increase in the matching size. The increase stops with only three number of applications, with which the fraction of covered hospitals is  $1 - 1/e^3 \approx 0.95$ . From this point forward the negative effect takes over and because of the random allocation of hospitals some doctors receive multiple offers while others receive none.

The following example considers the extreme case where doctors apply to all hospitals. Interestingly, the size of the matching in this case equals to the case where doctors were allowed to apply to a single hospital.

► **Example 9** (One application equals  $n$  applications.). With  $n$  applications for each doctor and hospitals accepting one interview, the social welfare of the matching approaches  $(1 - 1/e) \approx 0.63$  in a large market.

**Proof.** Since hospital preferences are uniformly random, each hospital selects an applicant to make an offer to, uniformly at random. Because doctors have  $n$  applications, the set of doctors applying to each hospital is the set of all doctors. Therefore the number of doctors who receive an offer is:

$$\lim_{n \rightarrow \infty} 1 - (1 - \frac{1}{n})^n = 1 - \frac{1}{e}$$

◀

### 3.2 Fractional Expected Number of Applications: A Scallop-Shape Function

Unlike Section 3.1, where there was a universal limit on the number of applications by doctors, in this subsection we study the social welfare when doctors are allowed to send different number of applications. First we show that for any expected number of applications  $x$ , the optimal social welfare occurs when doctors send either  $\lceil x \rceil$  or  $\lfloor x \rfloor$  applications. Then we study the social welfare as a function of the expected number of applications. We observe that granting extra applications to a small set of doctors and also retracting an application from a small set both hurt the market; suggesting that unfair treatment is not efficient in terms of social welfare.

► **Theorem 10.** *The social welfare of the matching with expected number of applications  $k < x < k + 1$ , when applicants either apply to  $k$  or  $k + 1$  positions is:*

$$(\lceil x \rceil - x)(1 - (1 - \frac{(1 - e^{-x})}{x})^k) + (x - \lfloor x \rfloor)(1 - (1 - \frac{(1 - e^{-x})}{x})^{k+1});$$

*this function is illustrated in Figure 2.*

The following proposition shows that for any expected number of applications  $x$ , the optimal social welfare occurs when doctors send either  $\lceil x \rceil$  or  $\lfloor x \rfloor$  applications.

► **Proposition 11.** *With any expected number of applications,  $x$ , the distribution of number of applications that achieves the highest social welfare, is one that allocates  $\lfloor x \rfloor$  applications to some doctors and  $\lceil x \rceil$  to the others, such that the expected number of applications equals  $x$ .*

**Proof.** Suppose the applications are distributed in a different way. Therefore, there are two doctors with  $l$  and  $k$  applications such that  $k - l \geq 2$ . We show that the size of matching is improved if we allocate  $f = \lfloor (k + l)/2 \rfloor$  and  $c = \lceil (k + l)/2 \rceil$  applications to those doctors. This alteration does not change the probability of receiving offers by other doctors as the applications are independent from doctors' perspective. So the only difference is the probability of receiving offers by these two doctors. Let  $p$  be the probability of a random application to lead to an offer. We claim

$$1 - (1 - p)^k + 1 - (1 - p)^l \leq 1 - (1 - p)^c + 1 - (1 - p)^f.$$

Since  $(1 - p)^c$  and  $(1 - p)^f$  have the same product as  $(1 - p)^k$  and  $(1 - p)^l$ , their sum is larger when the two factors are far apart, therefore:

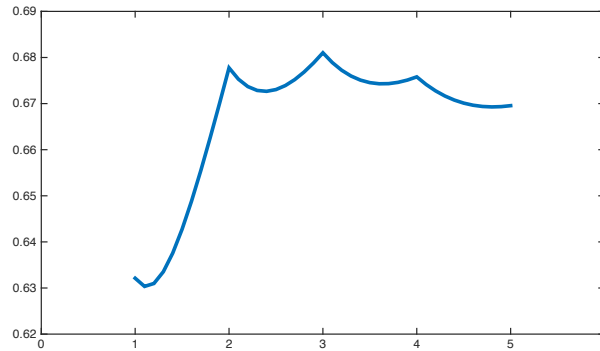
$$(1 - p)^c + (1 - p)^f \leq (1 - p)^k + (1 - p)^l$$

which implies the conclusion. ◀

► **Remark.** This argument shows that in order to find the optimal social welfare for different expected number of applications, we only need to study the case where doctors apply to the same number of hospitals –as studied in Section 3.1– or two consecutive numbers.

**Proof of Theorem 10.** This proof is similar to that of Proposition 6. From a doctor's perspective, the probability of a random application, leading to an offer is  $c/\mathcal{A}$ , where  $c$  is the number of covered hospitals, and  $\mathcal{A}$  is the number of applications. With  $nx$  applications, the fraction of covered hospitals equals  $(1 - e^{-x})$ . Thus, the probability of an application turning to an offer equals  $(1 - e^{-x})/x$ . Therefore, the expected social welfare of the matching which is the same as the probability of a random doctor receiving an offer is:

$$(\lceil x \rceil - x)(1 - (1 - \frac{(1 - e^{-x})}{x})^k) + (x - \lfloor x \rfloor)(1 - (1 - \frac{(1 - e^{-x})}{x})^{k+1}).$$
◀



■ **Figure 2** Size of matching with respect to expected number of applications, when hospitals make only one offer.

The scallop-shape Figure 2 illustrates the size of matching as a function of expected number of applications. The unusual behavior of the function at integer points shows that allowing a small group to apply to one more position, or limiting the number of applications of a small group to one less application, has a negative effect on size of matching.

► **Observation 12** (3 applications per doctor is the most efficient). *Theorem 10, depicted in Figure 6, and Proposition 11 imply that for any expected number of applications per doctor, and any independent distribution of those applications to doctors, the efficiency of the matching never exceeds  $\approx 0.68$ . This efficiency is achieved when all doctors apply to 3 hospitals. This is in sharp contrast with unlimited number of interviews (Proposition 5).*

#### 4 Granting multiple interviews

In this section we describe our approach for the general case. The key simplifying feature of the single-offer case was that each hospital immediately made an offer to their top applicant and each doctor was matched to their most preferred hospital that they received an offer from. However, with multiple interviews, finding the final matching requires actually running the usual procedure of the deferred acceptance algorithm.

The main result of this section is the following.

► **Theorem 13.** *Suppose each doctor has  $k$  applications and each hospital grants interviews to at most  $k'$  doctors. The social welfare of the matching in this case is  $1 - (1 - p)^k$  where  $0 < p < 1$  is the solution to*

$$1 - (1 - p)^k = \sum_{i=1}^{k'-1} [g(i, k, p)f(i; k)] + g(k', k, p)(1 - F(k' - 1; k));$$

where  $g(i, k, p) = 1 - (1 - \frac{1-(1-p)^k}{pk})^i$ ,  $f(i; k) = \frac{k^i e^{-k}}{i!}$  is the pmf of Poisson distribution with mean  $k$ , and  $F(\cdot; k)$  is the CDF of that distribution. Also when the expected number of applications is  $x$  such that  $x - \lfloor x \rfloor = z$ , the social welfare is upperbounded by  $(1 - z)(1 - (1 - p)^{\lfloor x \rfloor}) + z(1 - (1 - p)^{\lceil x \rceil})$ , where  $0 < p < 1$  is the solution to

$$(1 - z)(1 - (1 - p)^k) + z(1 - (1 - p)^{k+1}) = \sum_{i=1}^{k'-1} [h(i, x, p)f(i; x)] + h(k', x, p)(1 - F(k' - 1; x));$$

and  $h(i, x, p) = (1 - (1 - \frac{(1-(x-\lfloor x \rfloor))(1-(1-p)^{\lfloor x \rfloor}) + (x-\lfloor x \rfloor)(1-(1-p)^{\lceil x \rceil})}{px})^i)$ . The upperbound is tight when all doctors apply to either  $k$  or  $k + 1$  positions.

Before proceeding with the proof we give a high-level overview of the main steps. In a market of two equal sides the probability of a doctor being matched is equal to the probability of a hospital being matched. We use this equation to find the size of matching. The formula for probability of a doctor being matched is similar to the previous section; however, formulating the probability of a hospital being matched is more complicated. The main technical portion of this section is devoted to formulating this probability. First, we define a modified implementation of the matching procedure which results in the same outcome as the original implementation, but is easier to deal with for analytic purposes. The new implementation introduces a concept called “semi-proposal”. Later, we find the relationship between the number of applications, valid applications, proposals, and semi-proposals. The probability of a hospital being matched can be formulated in terms of the number of proposals. Finally, by finding the three other quantities, we find the number of proposals, and formulate the probability of a hospital being matched.

**Modified Implementation of Matching Procedure.** For the sake of analysis, it is useful to define a *modified implementation* of the matching procedure. Similar to the original implementation, in the first stage doctors apply to hospitals using all their applications and

hospitals conduct interviews with a subset of their applicants. The difference between the two implementations arises in the second stage. In the modified implementation, doctors submit their ordered preferences including *both* the hospitals they interviewed and the hospitals that rejected them for interview. Therefore, when the system simulates doctor-proposing deferred acceptance algorithm, doctors are allowed to propose using any of their applications; both valid and invalid ones. Since the invalid applications do not exist in hospitals' lists they get rejected immediately and the outcomes of both implementations are the same. In contrast to the modified implementation, we refer to the main procedure previously defined as the *original* implementation.

**Semi-proposal.** The procedure in the second stage of the modified implementation that simulates doctors proposing to hospitals as steps of the deferred acceptance algorithm. Note that this includes both the proposals that do not really count (because the hospital didn't extend an interview) and those that led to an interview.

► **Lemma 14.** *The expected number of semi-proposals made by a doctor is  $\frac{1-(1-p)^k}{p}$ , where  $p$  is the probability of a random application turning to a permanent match.*

**Proof.** By Proposition 4 from a doctor's perspective each hospital they propose to is available to them with equal probability, and their availabilities are independent<sup>2</sup>. Consider the modified implementation defined above. In the second stage of the game doctors send semi-proposals to hospitals. Let  $p$  be the probability of a random semi-proposal becoming a permanent match. By independence the probability that a doctor becomes matched is  $1 - (1 - p)^k$ . Also the expected number of semi-proposals made by a doctor is  $1 + (1 - p) + \dots + (1 - p)^{k-1} = \frac{1-(1-p)^k}{p}$ . ◀

► **Lemma 15.** *For any application  $\mathcal{A} = (d, h)$ , consider  $\mathcal{E}_{\text{semi}}(\mathcal{A})$  as the event that  $\mathcal{A}$  turns to a semi-proposal, i.e.,  $d$  sends a semi-proposal to  $h$ , and  $\mathcal{E}_{\text{valid}}(\mathcal{A})$  as the event where  $\mathcal{A}$  is valid.  $\mathcal{E}_{\text{semi}}(\mathcal{A})$  and  $\mathcal{E}_{\text{valid}}(\mathcal{A})$  are independent.*

**Proof.** Application  $\mathcal{A} = (d, h)$  is turned into a semi-proposal if the previous semi-proposals of  $d$  are rejected. Similarly, a valid application  $\mathcal{V}$  is turned into a proposal if the previous semi-proposals are rejected. Since preferences are distributed uniformly at random for both doctors and hospitals, the fact that  $\mathcal{A}$  is valid or invalid is independent of the rank of  $h$  in the preference list of  $d$ . Therefore, the probability of  $\mathcal{A}$  turning into a semi-proposal conditioned on it being valid is equal to the unconditional probability. ◀

► **Corollary 16.** *The probability of a random valid application turning into a proposal,  $p_{V \rightarrow P}$ , equals the probability of a random application into a semi-proposal,  $p_{A \rightarrow S}$ . This probability is  $\frac{\mathcal{P}}{\mathcal{A}} = \frac{\mathcal{S}}{\mathcal{V}}$ , where  $\mathcal{P}$ ,  $\mathcal{A}$ ,  $\mathcal{S}$ , and  $\mathcal{V}$  represent the number of proposals, applications, semi-proposals and valid applications, respectively.*

**Proof.** Lemma 15 implies  $p_{V \rightarrow P} = p_{A \rightarrow S}$ . The first probability equals the ratio of total number of proposals to valid applications, and the second equals the total number of semi-proposals to applications. ◀

A key observation for finding the number of valid applications is the following.

<sup>2</sup> Note that considering proposals *with replacement* is accurate up to lower order terms; with constant proposals, a doctor will make a repeat only with vanishingly small probability.

► **Observation 17.** *The number of applications that a hospital receives in the first stage forms a binomial distribution which converges to a Poisson distribution with mean  $k$  in the limit. This is due to the independent and uniform preferences of doctors.*

► **Lemma 18.** *The expected number of proposals that a hospital receives is*

$$\frac{1 - (1 - p)^k}{p} \cdot \frac{\sum_{i=1}^{k'-1} \left[ i \cdot \frac{k^i e^{-k}}{i!} \right] + k'(1 - e^{-k} \sum_{i=0}^{k'-1} \frac{k^i}{i!})}{k}.$$

**Proof.** By Corollary 16,  $\frac{\mathcal{P}}{\mathcal{A}} = \frac{\mathcal{S}}{\mathcal{V}}$ . In order to find  $\mathcal{P}$ , we need  $\mathcal{A}$ ,  $\mathcal{S}$ , and  $\mathcal{V}$ . By definition,  $\mathcal{A} = k$  and by Lemma 14,  $\mathcal{S} = \frac{1 - (1 - p)^k}{p}$ . Therefore, we only need to find  $\mathcal{V}$ . Recall that an application  $(d, h)$  is valid if hospital  $h$  does not reject it in the first stage. A hospital rejects an application only if it receives more than  $k'$ . Therefore, by Observation 17, the expected number of valid applications per hospital is,

$$\mathcal{V} = \sum_{i=1}^{k'-1} i \cdot \frac{k^i e^{-k}}{i!} + k'(1 - e^{-k} \sum_{j=0}^{k'-1} \frac{k^j}{j!}).$$

Substituting  $\mathcal{A}$ ,  $\mathcal{S}$ , and  $\mathcal{P}$  in Corollary 16, the expected number of proposals a hospital receives is

$$\frac{1 - (1 - p)^k}{p} \cdot \frac{\sum_{i=1}^{k'-1} i \cdot \frac{k^i e^{-k}}{i!} + k'(1 - e^{-k} \sum_{i=0}^{k'-1} \frac{k^i}{i!})}{k}. \quad \blacktriangleleft$$

Now we find the probability that a random hospital is matched.

► **Lemma 19.** *The probability of a random hospital being matched is*

$$\sum_{i=1}^{k'-1} [g(i, k, p) f(i; k)] + g(k', k, p)(1 - F(k' - 1; k));$$

where  $g(i, k, p) = 1 - (1 - \frac{1 - (1 - p)^k}{pk})^i$ ,  $f(i; k) = \frac{k^i e^{-k}}{i!}$  is the pmf of Poisson distribution with mean  $k$ ,  $F(\cdot; k)$  is the CDF of that distribution, and  $p$  is the probability of a random application turning to a permanent match.

**Proof.** From the hospitals' point of view, each of their valid applications has the same probability of becoming a proposal. Let  $q$  be the probability of a random valid application turning into a proposal. In a doctors-proposing deferred-acceptance algorithm, if a hospital is once tentatively matched it will remain matched forever. A hospital becomes tentatively matched if it receives a proposal, in other words if at least one of its valid applications becomes a proposal. The probability that a random hospital with  $j$  valid applications receives a proposal is  $1 - (1 - q)^j$ . Therefore, the probability that a random hospital is matched equals,

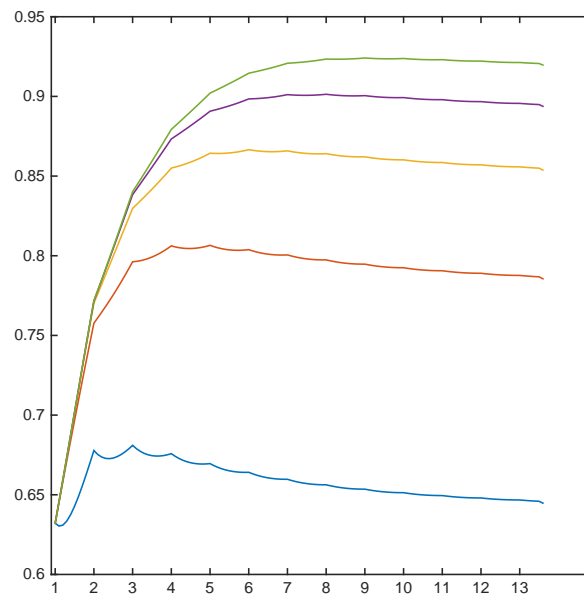
$$\sum_{i=1}^{k'-1} \left[ (1 - (1 - q)^i) \frac{k^i e^{-k}}{i!} \right] + (1 - (1 - q)^{k'}) (1 - e^{-k} \sum_{i=0}^{k'-1} \frac{k^i}{i!}).$$

The probability of a random valid application turning into a proposal,  $q$ , is equal to the ratio of the total number of proposals to the total number of valid applications. By Lemma 15, this ratio is equal to the ratio of the total number of semi-proposals to the total number of applications. Therefore,  $q = \frac{1 - (1 - p)^k}{pk}$ . ◀

**Proof of Theorem 13.** Lemma 14, Lemma 19 and the fact that in a matching of two equal sides, the probability of a random doctor being matched is equal to a hospital being matched implies:

$$1 - (1 - p)^k = \sum_{i=1}^{k'-1} [g(i, k, p)f(i; k)] + g(k', k, p)(1 - F(k' - 1; k)).$$

Similar reasoning holds when the expected number of applications is an arbitrary real number  $x$  such that each doctor has either  $\lfloor x \rfloor$  or  $\lceil x \rceil$  applications, deriving the formula for general  $x$ . By Proposition 11, the size of the matching with expected  $k \leq x < k + 1$  applications is maximized when all doctors apply to either  $k$  or  $k + 1$  positions. This concludes the proof. ◀



**Figure 3** Size of matching with respect to expected number of application. The curves represents the settings where hospitals allow one to five interviews. The lowest curve belongs to one interview and the highest curve to five interviews.

## 5 Extensions

In this section we discuss two extensions to our main results in Section 4. Section 5.1 discusses the extension to unbalanced markets where the number of doctors and positions is different. Section 5.2 discusses the extension to correlated preferences.

### 5.1 Beyond Balanced Markets

In this part, we show that the phenomenon of the positive effect of setting the same limit for all doctors is not limited to a balanced market – where the number of applicants and positions are the same – but extends to unbalanced markets with different number of applicants and



positions. Although the same phenomenon exists more generally, the exact function is not preserved and the optimum number of applications depends on the ratio between the sizes of the two sides.

We study the case where hospitals make one offer (similar to Section 3) and the ratio of the number of hospitals to the number of doctors is  $r$ . The results from Section 4 with multiple interviews also generalize to this setting. However, for simplicity and similarity of the outcomes, we just present the results for the model with one offer. Similar to the previous section, we can compute the size of the matching as a function of the expected number of applications, when doctors send  $k$  or  $k + 1$  applications.

► **Proposition 20.** *The expected fraction of matched doctors with expected number of applications  $k < x < k + 1$ , when applicants either apply to  $k$  or  $k + 1$  positions and when the ratio of number of hospitals to number of doctors is  $r$  is:*

$$(\lceil x \rceil - x)(1 - (1 - \frac{r(1 - e^{-x/r})}{x})^k) + (x - \lfloor x \rfloor)(1 - (1 - \frac{r(1 - e^{-x/r})}{x})^{k+1}).$$

**Proof.** As shown in the proof of Proposition 6, from the doctors perspective, the probability of a random application, leading to an offer is  $\frac{\mathcal{C}}{\mathcal{A}}$ , where  $\mathcal{C}$  is the number of covered hospitals and  $\mathcal{A}$  is the number of application. From the hospitals perspective, each application is equally likely to be sent to each hospital. Therefore the number of applications received is distributed as a Poisson distribution with  $\lambda = \frac{x}{r}$ . So the number of expected covered hospitals is  $rn(1 - e^{-x/r})$  and the probability of a random application, leading to an offer is  $\frac{r(1 - e^{-x/r})}{x}$ . Therefore the expected fraction of doctors who are matched which is the same as the probability of a random doctor receiving an offer is:

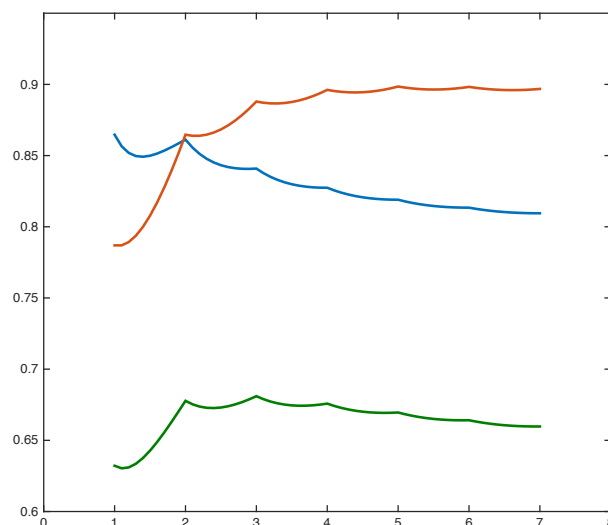
$$(\lceil x \rceil - x)(1 - (1 - \frac{r(1 - e^{-x/r})}{x})^k) + (x - \lfloor x \rfloor)(1 - (1 - \frac{r(1 - e^{-x/r})}{x})^{k+1}). \quad \blacktriangleleft$$

Based on Definition 3, the social welfare of a matching is the ratio of the size of the matching to the size of maximum matching; and in unbalanced markets, the maximum size is the size of the smaller side of the matching. For  $r \geq 1$ , the doctors make the smaller side therefore the social welfare is equal to expected fraction of doctors who are matched as computed in Proposition 20. If  $r < 1$ , the social welfare is the expected fraction of doctors who are matched as computed in Proposition 20 divided by  $r$ .

Section 5.1 shows the social welfare of the matching as a function of expected number of applications for  $r = \frac{1}{2}, 1, 2$ . In the figure, the red function refers to  $r = 2$ , the blue function to  $r = 1/2$  and the green function to  $r = 1$ .

As seen in the figure a similar structure holds for unbalanced networks, but the optimal number of applications depends on the factor of balancedness  $r$ . When the number of hospitals is half of the number of doctors, the market achieves its maximum size when each doctor applies to just one position. With more applications, hospitals become more congested; and the number of covered hospitals and therefore the number of total offers does not increase significantly. Therefore, the rejection probability of applications increases and with higher probability a doctor remains unmatched. In contrast, when the number of hospitals is more than the number of doctors, allowing more applications has a positive effect. With more applications –while the number of applications is still small– more hospitals receive at least one application; therefore, the number of offers increases considerably.

Also, note that in the figure, the social welfare of the balanced market is generally lower than the social welfare of markets with  $r = 1/2, 2$ . This is not surprising since by definition the social welfare is the fraction of matched individuals on the smaller side of the market:



■ **Figure 4** Social welfare of the matching with respect to the expected number of applications. The red curve refers to  $r = 2$ , blue curve to  $r = 1/2$  and the green curve to  $r = 1$ , where  $r$  is the ratio of the number of hospitals to doctors.

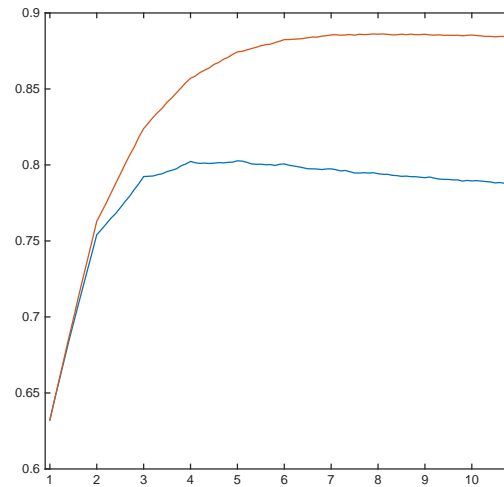
doctors for  $r \leq 1$ , and hospitals for  $r > 1$ . For  $r = 2$ , with the same number of applications per doctor, more hospitals are covered that leads to more offers and a higher fraction of matched doctors. For  $r = 1/2$ , with the same number of applications, a higher fraction of hospitals is covered which leads to a higher fraction of matched hospitals.

## 5.2 Beyond Uniform Preferences

In this part, we go beyond the uniform and independent assumptions and use simulations to show that the properties of limited interviews exist more generally. The model studied in the previous sections assumes that doctors and hospitals preferences remain uniform and independent after the interview stage. We relax this assumption. We study a model where the two sides have no information prior to the matching procedure. However, after the interview stage, all participants refine their lists with information learned in the interviews. Therefore, preference lists may become correlated.

Without prior knowledge about the other side, the behaviors in the first stage is similar to what stated previously: there is a Bayes Nash Equilibrium such that doctors and hospitals pick the top of their lists. In the second stage, doctors and hospitals are not anymore identical in terms of popularity. We will be assuming that preferences among the interviews are independent samples from a distribution that reflects popularity. Immorlica and Mahdian [6] show that in a large market short preference lists sampled from a distribution, all participants are likely to prefer to be truthful. While this is not exactly the same as our model, analogous to this case, we will make the assumption in our simulation that participants do act truthfully.

To show how generally the properties hold, for simulations we study the other extreme in preferences: completely aligned preferences after interviews. We use a market of 10,000 doctors and hospitals. Similar to previous sections we start with uniform and independent preference in the first stage. For the second stage we simulate running the deferred acceptance algorithm when hospitals agree on a random preference over doctors. Figure 5 shows the size of matching as a function of number of applications when the number of interviews for hospitals is limited to two and four interviews.



■ **Figure 5** Size of matching with respect to expected number of application. In the first stage the preferences are uniform and independent. In the second stage hospitals preferences are aligned. The bottom curve belongs to two interviews and the top one to four interviews.

As observed in Figure 5, the properties of limited interviews are not restricted to uniform preferences. They hold even in the other extreme case where the preferences are completely correlated after interviews.

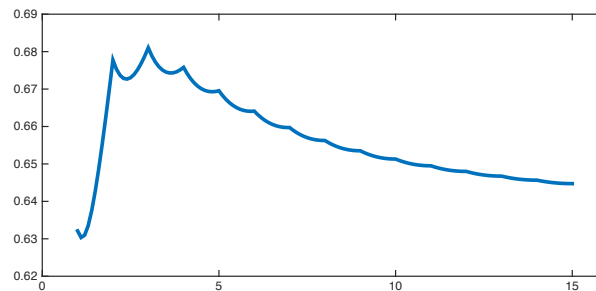
## References

- 1 Nick Arnosti. Centralized clearinghouse design: A quantity-quality tradeoff. *Working Paper*, 2016.
- 2 Christopher Avery and Jonathan Levin. Early admissions at selective colleges. *The American Economic Review*, 100(5):2125–2156, 2010.
- 3 Hedyeh Beyhaghi, Daniela Saban, and Eva Tardos. Effect of selfish choices in deferred acceptance with short lists. *Match-Up*, 2017.
- 4 Joanna Drummond, Allan Borodin, and Kate Larson. Natural interviewing equilibria for stable matching. *working paper*, 2016.
- 5 D. Gale and L. S. Shapley. College Admissions and the Stability of Marriage. *American Mathematical Monthly*, 69(1):9–15, 1962.
- 6 Nicole Immorlica and Mohammad Mahdian. Marriage, honesty, and stability. In *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 53–62 (electronic). ACM, New York, 2005.
- 7 Sangram Vilasrao Kadam. Interviewing in matching markets. *Working Paper*, 2015.
- 8 Fuhito Kojima and Parag A. Pathak. Incentives and stability in large two-sided matching markets. *American Economic Review*, 99(3):608–27, 2009. doi:10.1257/aer.99.3.608.
- 9 Robin S Lee and Michael Schwarz. Interviewing in two-sided matching markets. Working Paper 14922, National Bureau of Economic Research, April 2009. doi:10.3386/w14922.
- 10 Alvin E Roth and Marilda Sotomayor. Two-sided matching. *Handbook of game theory with economic applications*, 1:485–541, 1992.

## A Missing Proofs

**Discussion of Observation 2.** The truthful equilibrium is due to the independence and symmetry and the private information model (the doctors and hospitals are unaware of the number of applications and preference lists of others). At all times, no hospital or doctor is more popular than the rest. Therefore from a doctor's point of view the probability of acceptance in all hospitals is the same and there is no benefit in not applying to a favorite hospital or be non-truthful about the preference order. The same holds for hospitals. ◀

**Proof of Proposition 5.** We show that if one of the doctors who previously had  $k$  applications, now has  $k + 1$  applications and the number of applications of other doctors remain the same, the size of the matching can only increase. By Observation 2, both doctors and hospitals are truthful, therefore we are comparing the size of matching as the result of deferred acceptance algorithm when doctor  $d$  has an extra application and all other doctors have the same number of applications. Since the deferred acceptance algorithm is oblivious to the order in which doctors proposes, we may hold out the last application of doctor  $d$  and find the outcome when the doctor  $d$  does not have this application in his/her list. The result of the deferred acceptance algorithm, without this application is the same as the case where doctor  $d$  had  $k$  applications. We show that this last application can only increase the number of matching. If doctor  $d$  is matched with one of his/her first  $k$  proposals, the last application does not change. If doctor  $d$  proposes to  $k + 1_{st}$  hospital, it can only increase the number of hospitals who have received any proposal. ◀



■ **Figure 6** Size of matching with respect to expected number of applications, when hospitals make only one offer. (Extended version of Figure 2.)

# Counterexamples to the Low-Degree Conjecture

**Justin Holmgren**

NTT Research, Palo Alto, CA, USA  
justin.holmgren@ntt-research.com

**Alexander S. Wein**

Courant Institute of Mathematical Sciences, New York University, NY, USA  
awein@cims.nyu.edu

---

## Abstract

A conjecture of Hopkins (2018) posits that for certain high-dimensional hypothesis testing problems, no polynomial-time algorithm can outperform so-called “simple statistics”, which are low-degree polynomials in the data. This conjecture formalizes the beliefs surrounding a line of recent work that seeks to understand statistical-versus-computational tradeoffs via the *low-degree likelihood ratio*. In this work, we refute the conjecture of Hopkins. However, our counterexample crucially exploits the specifics of the noise operator used in the conjecture, and we point out a simple way to modify the conjecture to rule out our counterexample. We also give an example illustrating that (even after the above modification), the symmetry assumption in the conjecture is necessary. These results do not undermine the low-degree framework for computational lower bounds, but rather aim to better understand what class of problems it is applicable to.

**2012 ACM Subject Classification** Theory of computation → Computational complexity and cryptography

**Keywords and phrases** Low-degree likelihood ratio, error-correcting codes

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.75

**Funding** *Justin Holmgren*: Most of this work was done while with the Simons Institute for the Theory of Computing.

*Alexander S. Wein*: Partially supported by NSF grant DMS-1712730 and by the Simons Collaboration on Algorithms and Geometry.

**Acknowledgements** We thank Sam Hopkins and Tim Kunisky for comments on an earlier draft.

## 1 Introduction

A primary goal of computer science is to understand which problems can be solved by efficient algorithms. Given the formidable difficulty of proving unconditional computational hardness, state-of-the-art results typically rely on unproven conjectures. While many such results rely only upon the widely-believed conjecture  $P \neq NP$ , other results have only been proven under stronger assumptions such as the unique games conjecture [19, 20], the exponential time hypothesis [16], the learning with errors assumption [25], or the planted clique hypothesis [17, 4].

It has also been fruitful to conjecture that a specific *algorithm* (or limited class of algorithms) is optimal for a suitable class of problems. This viewpoint has been particularly prominent in the study of average-case noisy statistical inference problems, where it appears that optimal performance over a large class of problems can be achieved by methods such as the sum-of-squares hierarchy (see [24]), statistical query algorithms [18, 5], the approximate message passing framework [9, 22], and low-degree polynomials [15, 14, 13]. It is helpful to have such a conjectured-optimal meta-algorithm because this often admits a *systematic* analysis of hardness. However, the exact class of problems for which we believe these methods



© Justin Holmgren and Alexander S. Wein;  
licensed under Creative Commons License CC-BY  
12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 75; pp. 75:1–75:9



Leibniz International Proceedings in Informatics

LIPIcs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

are optimal has typically not been precisely formulated. In this work, we explore this issue for the class of low-degree polynomial algorithms, which admits a systematic analysis via the low-degree likelihood ratio.

The *low-degree likelihood ratio* [15, 14, 13] has recently emerged as a framework for studying computational hardness in high-dimensional statistical inference problems. It has been shown that for many “natural statistical problems,” all known polynomial-time algorithms only succeed in the parameter regime where certain “simple” (low-degree) statistics succeed. The power of low-degree statistics can often be understood via a relatively simple explicit calculation, yielding a tractable way to precisely predict the statistical-versus-computational tradeoffs in a given problem. These “predictions” can rigorously imply lower bounds against a broad class of spectral methods [21, Theorem 4.4] and are intimately connected to the *sum-of-squares hierarchy* (see [14, 13, 24]). Recent work has (either explicitly or implicitly) carried out this type of low-degree analysis for a variety of statistical tasks [3, 15, 14, 13, 2, 1, 21, 8, 6, 23, 7]. For more on these methods, we refer the reader to the PhD thesis of Hopkins [13] or the survey article [21].

Underlying the above ideas is the belief that for certain “natural” problems, low-degree statistics are as powerful as all polynomial-time algorithms – we refer broadly to this belief as the “low-degree conjecture”. However, formalizing the notion of “natural” problems is not a straightforward task. Perhaps the easiest way to illustrate the meaning of “natural” is by example: prototypical examples (studied in the previously mentioned works) include planted clique, sparse PCA, random constraint satisfaction problems, community detection in the stochastic block model, spiked matrix models, tensor PCA, and various problems of a similar flavor. All of these can be stated as simple hypothesis testing problems between a “null” distribution (consisting of random noise) and a “planted” distribution (which contains a “signal” hidden in noise). They are all high-dimensional problems but with sufficient symmetry that they can be specified by a small number of parameters (such as a “signal-to-noise ratio”). For all of the above problems, the best known polynomial-time algorithms succeed precisely in the parameter regime where simple statistics succeed, i.e., where there exists a  $O(\log n)$ -degree polynomial of the data whose value behaves noticeably different under the null and planted distributions (in a precise sense). Thus, barring the discovery of a drastically new algorithmic approach, the low-degree conjecture seems to hold for all the above problems. In fact, a more general version of the conjecture seems to hold for runtimes that are not necessarily polynomial: degree- $D$  statistics are as powerful as all  $n^{\tilde{\Theta}(D)}$ -time algorithms, where  $\tilde{\Theta}$  hides factors of  $\log n$  [13, Hypothesis 2.1.5] (see also [21, 8]).

A precise version of the low-degree conjecture was formulated in the PhD thesis of Hopkins [13]. This includes precise conditions on the null distribution  $\nu$  and planted distribution  $\mu$  which capture most of the problems mentioned above. The key conditions are that there should be sufficient symmetry, and that  $\mu$  should be injected with at least a small amount of noise. Most of the problems above satisfy this symmetry condition (a notable exception being the spiked Wishart model<sup>1</sup>, which satisfies a mild generalization of it), but it remained unclear whether this assumption was needed in the conjecture. On the other hand, the noise assumption is certainly necessary, as illustrated by the example of solving a system of linear equations over a finite field: if the equations have an exact solution then it can be obtained via Gaussian elimination even though low-degree statistics suggest that the

---

<sup>1</sup> Here we mean the formulation of the spiked Wishart model used in [1], where we directly observe Gaussian samples instead of only their covariance matrix.

problem should be hard; however, if a small amount of noise is added (so that only a  $1 - \varepsilon$  fraction of the equations can be satisfied) then Gaussian elimination is no longer helpful, and the low-degree conjecture seems to hold.

In this work we investigate more precisely what kinds of noise and symmetry conditions are needed in the conjecture of Hopkins [13]. Our first result (Theorem 4) actually refutes the conjecture in the case where the underlying random variables are real-valued. Our counterexample exploits the specifics of the noise operator used in the conjecture, along with the fact that a single real number can be used to encode a large (but polynomially bounded) amount of data. In other words, we show that a *stronger* noise assumption than the one in [13] is needed; Remark 6 explains a modification of the conjecture that we do not know how to refute. Our second result (Theorem 7) shows that the symmetry assumption in [13] cannot be dropped, i.e., we give a counterexample for a weaker conjecture that does not require symmetry. Both of our counterexamples are based on efficiently decodable error-correcting codes.

## Notation

Asymptotic notation such as  $o(1)$  and  $\Omega(1)$  pertains to the limit  $n \rightarrow \infty$ . We say that an event occurs with *high probability* if it occurs with probability  $1 - o(1)$ , and we use the abbreviation w.h.p. (“with high probability”). We use  $[n]$  to denote the set  $\{1, 2, \dots, n\}$ . The *Hamming distance* between vectors  $x, y \in F^n$  (for some field  $F$ ) is  $\Delta(x, y) = |\{i \in [n] : x_i \neq y_i\}|$  and the *Hamming weight* of  $x$  is  $\Delta(x, 0)$ .

## 2 The Low-Degree Conjecture

We now state the formal variant of the low-degree conjecture proposed in the PhD thesis of Hopkins [13, Conjecture 2.2.4]. The terminology used in the statement will be explained below.

► **Conjecture 1.** *Let  $\mathcal{X}$  be a finite set or  $\mathbb{R}$ , and let  $k \geq 1$  be a fixed integer. Let  $N = \binom{n}{k}$ . Let  $\nu$  be a product distribution on  $\mathcal{X}^N$ . Let  $\mu$  be another distribution on  $\mathcal{X}^N$ . Suppose that  $\mu$  is  $S_n$ -invariant and  $(\log n)^{1+\Omega(1)}$ -wise almost independent with respect to  $\nu$ . Then no polynomial-time computable test distinguishes  $T_\delta \mu$  and  $\nu$  with probability  $1 - o(1)$ , for any  $\delta > 0$ . Formally, for all  $\delta > 0$  and every polynomial-time computable  $t : \mathcal{X}^N \rightarrow \{0, 1\}$  there exists  $\delta' > 0$  such that for every large enough  $n$ ,*

$$\frac{1}{2} \mathbb{P}_{x \sim \nu}(t(x) = 0) + \frac{1}{2} \mathbb{P}_{x \sim T_\delta \mu}(t(x) = 1) \leq 1 - \delta'.$$

We now explain some of the terminology used in the conjecture, referring the reader to [13] for the full details. We will be concerned with the case  $k = 1$ , in which case  $S_n$ -invariance of  $\mu$  means that for any  $x \in \mathcal{X}^n$  and any  $\pi \in S_n$  (the symmetric group) we have  $\mathbb{P}_\mu(x) = \mathbb{P}_\mu(\pi \cdot x)$  where  $\pi$  acts by permuting coordinates. The notion of *D-wise almost independence* captures how well degree- $D$  polynomials can distinguish  $\mu$  and  $\nu$ . For our purposes, we do not need the full definition of *D-wise almost independence* (see [13]), but only the fact that it is implied by *exact D-wise independence*, defined as follows.

► **Definition 2.** *A distribution  $\mu$  on  $\mathcal{X}^N$  is  $D$ -wise independent with respect to  $\nu$  if for any  $S \subseteq [N]$  with  $|S| \leq D$  we have equality of the marginal distributions  $\mu|_S = \nu|_S$ .*

Finally, the *noise operator*  $T_\delta$  is defined as follows.



► **Definition 3.** Let  $\nu$  be a product distribution on  $\mathcal{X}^N$  and let  $\mu$  be another distribution on  $\mathcal{X}^N$ . For  $\delta \in [0, 1]$ , let  $T_\delta\mu$  be the distribution on  $\mathcal{X}^N$  generated as follows. To sample  $z \sim T_\delta\mu$ , first sample  $x \sim \mu$  and  $y \sim \nu$  independently, and then, independently for each  $i$ , let

$$z_i = \begin{cases} x_i & \text{with probability } 1 - \delta, \\ y_i & \text{with probability } \delta. \end{cases}$$

(Note that  $T_\delta$  depends on  $\nu$  but the notation suppresses this dependence;  $\nu$  will always be clear from context.)

### 3 Main Results

We first give a counterexample that refutes Conjecture 1 in the case  $\mathcal{X} = \mathbb{R}$ .

► **Theorem 4.** *The following holds for infinitely many  $n$ . Let  $\mathcal{X} = \mathbb{R}$ , and  $\nu = \text{Unif}([0, 1]^n)$ . There exists a distribution  $\mu$  on  $\mathcal{X}^n$  such that  $\mu$  is  $S_n$ -invariant (with  $k = 1$ ) and  $\Omega(n)$ -wise independent with respect to  $\nu$ , and for some constant  $\delta > 0$  there exists a polynomial-time computable test distinguishing  $T_\delta\mu$  and  $\nu$  with probability  $1 - o(1)$ .*

The proof is given in Section 5.1.

► **Remark 5.** We assume a standard model of finite-precision arithmetic over  $\mathbb{R}$ , i.e., the algorithm  $t$  can access polynomially-many bits in the binary expansion of its input.

Note that we refute an even weaker statement than Conjecture 1 because our counterexample has *exact*  $\Omega(n)$ -wise independence instead of only  $(\log n)^{1+\Omega(1)}$ -wise *almost* independence.

► **Remark 6.** Essentially, our counterexample exploits the fact that a single real number can be used to encode a large block of data, and that the noise operator  $T_\delta$  will leave many of these blocks untouched (effectively allowing us to use a super-constant alphabet size). We therefore propose modifying Conjecture 1 in the case  $\mathcal{X} = \mathbb{R}$  by using a different noise operator that applies a small amount of noise to *every* coordinate instead of resampling a small number of coordinates. If  $\nu$  is i.i.d.  $\mathcal{N}(0, 1)$  then the standard Ornstein-Uhlenbeck noise operator is a natural choice (and in fact, this is mentioned by [13]). Formally, this is the noise operator  $T_\delta$  that samples  $T_\delta\mu$  as follows: draw  $x \sim \mu$  and  $y \sim \nu$  and output  $\sqrt{1 - \delta}x + \sqrt{\delta}y$ .

Our second result illustrates that in the case where  $\mathcal{X}$  is a finite set, the  $S_n$ -invariance assumption cannot be dropped from Conjecture 1. (In stating the original conjecture, Hopkins [13] remarked that he was not aware of any counterexample when the  $S_n$ -invariance assumption is dropped).

► **Theorem 7.** *The following holds for infinitely many  $n$ . Let  $\mathcal{X} = \{0, 1\}$  and  $\nu = \text{Unif}(\{0, 1\}^n)$ . There exists a distribution  $\mu$  on  $\mathcal{X}^n$  such that  $\mu$  is  $\Omega(n)$ -wise independent with respect to  $\nu$ , and for some constant  $\delta > 0$  there exists a polynomial-time computable test distinguishing  $T_\delta\mu$  and  $\nu$  with probability  $1 - o(1)$ .*

The proof is given in Section 5.2.

Both of our counterexamples are in fact still valid in the presence of a stronger noise operator  $T_\delta$  that *adversarially* changes any  $\delta$ -fraction of the coordinates.

The rest of the paper is organized as follows. Our counterexamples are based on error-correcting codes, so in Section 4 we review the basic notions from coding theory that we will need. In Section 5 we construct our counterexamples and prove our main results.

## 4 Coding Theory Preliminaries

Let  $F = \mathbb{F}_q$  be a finite field. A *linear code*  $C$  (over  $F$ ) is a linear subspace of  $F^n$ . Here  $n$  is called the (*block*) *length*, and the elements of  $C$  are called *codewords*. The *distance* of  $C$  is the minimum Hamming distance between two codewords, or equivalently, the minimum Hamming weight of a nonzero codeword.

► **Definition 8.** Let  $C$  be a linear code. The *dual distance* of  $C$  is the minimum Hamming weight of a vector in  $F^n$  that is orthogonal to all codewords. Equivalently, the dual distance is the distance of the dual code  $C^\perp = \{x \in F^n : \langle x, c \rangle = 0 \ \forall c \in C\}$ .

The following standard fact will be essential to our arguments.

► **Proposition 9.** If  $C$  is a linear code with dual distance  $d$ , then the uniform distribution over codewords is  $(d - 1)$ -wise independent with respect to the uniform distribution on  $F^n$ .

**Proof.** This is a standard fact in coding theory, but we give a proof here for completeness. Fix  $S \subseteq [n]$  with  $|S| \leq d - 1$ . For some  $k$ , we can write  $C = \{x^\top G : x \in F^k\}$  for some  $k \times n$  generator matrix  $G$  whose rows form a basis for  $C$ . Let  $G_S$  be the  $k \times |S|$  matrix obtained from  $G$  by keeping only the columns in  $S$ . It is sufficient to show that if  $x$  is drawn uniformly from  $F^k$  then  $x^\top G_S$  is uniform over  $F^{|S|}$ . The columns of  $G_S$  must be linearly independent, because otherwise there is a vector  $y \in F^n$  of Hamming weight  $\leq d - 1$  such that  $Gy = 0$ , implying  $y \in C^\perp$ , which contradicts the dual distance. Thus there exists a set  $T \subseteq [k]$  of  $|S|$  linearly independent rows of  $G_S$  (which form a basis for  $F^{|S|}$ ). For any fixed choice of  $x_{[k] \setminus T}$  (i.e., the coordinates of  $x$  outside  $T$ ), if the coordinates  $x_T$  are chosen uniformly at random then  $x^\top G_S$  is uniform over  $F^{|S|}$ . This completes the proof. ◀

► **Definition 10.** A code  $C$  admits *efficient decoding* from  $r$  errors and  $s$  erasures if there is a deterministic polynomial-time algorithm  $\mathcal{D} : (F \cup \{\perp\})^n \rightarrow F^n \cup \{\text{fail}\}$  with the following properties.

- For any codeword  $c \in C$ , let  $c' \in F^n$  be any vector obtained from  $c$  by changing the values of at most  $r$  coordinates (to arbitrary elements of  $F$ ), and replacing at most  $s$  coordinates with the erasure symbol  $\perp$ . Then  $\mathcal{D}(c') = c$ .
- For any arbitrary  $c' \in F^n$  (not obtained from some codeword as above),  $\mathcal{D}(c')$  can output any codeword or fail (but must never output a vector that is not a codeword<sup>2</sup>).

Note that the decoding algorithm knows where the erasures have occurred but does not know where the errors have occurred.

Our first counterexample (Theorem 4) is based on the classical *Reed-Solomon codes*  $\text{RS}_q(n, k)$ , which consist of univariate polynomials of degree at most  $k$  evaluated at  $n$  canonical elements of the field  $\mathbb{F}_q$ , and are known to have the following properties.

► **Proposition 11 (Reed-Solomon Codes).** For any integers  $0 \leq k < n$  and for any prime power  $q \geq n$ , there is a length- $n$  linear code  $C$  over  $\mathbb{F}_q$  with the following properties:

- the dual distance is  $k + 2$ ,
- $C$  admits efficient decoding from  $r$  errors and  $s$  erasures whenever  $2r + s < n - k$ .

**Proof.** See e.g., [11] for the construction and basic facts regarding Reed-Solomon codes  $\text{RS}_q(n, k)$ . The distance of  $\text{RS}_q(n, k)$  is  $n - k$ . It is well known that the dual code of  $\text{RS}_q(n, k)$  is  $\text{RS}_q(n, n - k - 2)$ , which proves the claim about dual distance. Efficient decoding is discussed e.g., in Section 3.2 of [11]. ◀

<sup>2</sup> The codes we deal with in this paper can be efficiently constructed, and so it is easy to test whether a given vector is a codeword. Thus, this assumption is without loss of generality.

Our second counterexample (Theorem 7) is based on the following construction of efficiently correctable binary codes with large dual distance. A proof (by Guruswami) can be found in [26, Theorem 4]. Similar results were proved earlier [27, 10, 12].

► **Proposition 12.** *There exists a universal constant  $\zeta \geq 1/30$  such that for every integer  $i \geq 1$  there is a linear code  $C$  over  $\mathbb{F}_2 = \{0, 1\}$  of block length  $n = 42 \cdot 8^{i+1}$ , with the following properties:*

- *the dual distance is at least  $\zeta n$ , and*
- *$C$  admits efficient decoding from  $\zeta n/2$  errors (with no erasures).*

## 5 Proofs

Before proving the main results, we state some prerequisite notation and lemmas.

► **Definition 13.** *Let  $F$  be a finite field. For  $S \subseteq [n]$  (representing erased positions), the  $S$ -restricted Hamming distance  $\Delta_S(x, y)$  is the number of coordinates in  $[n] \setminus S$  where  $x$  and  $y$  differ:*

$$\Delta_S(x, y) = |\{i \in [n] \setminus S : x_i \neq y_i\}|.$$

*We allow  $x, y$  to belong to either  $F^n$  or  $F^{[n] \setminus S}$ , or even to  $(F \cup \{\perp\})^n$  so long as the “erasures”  $\perp$  occur only in  $S$ .*

The following lemma shows that a random string is sufficiently far from any codeword.

► **Lemma 14.** *Let  $C$  be a length- $n$  linear code over a finite field  $F$ . Suppose  $C$  admits efficient decoding from  $2r$  errors and  $s$  erasures, for some  $r, s$  satisfying  $r \leq (n - s)/(8e)$ . For any fixed choice of at most  $s$  erasure positions  $S \subseteq [n]$ , if  $x$  is a uniformly random element of  $F^{[n] \setminus S}$  then*

$$\mathbb{P}_x(\exists c \in C : \Delta_S(c, x) \leq r) \leq (r + 1)2^{-r}.$$

**Proof.** Let  $B_r(c) = \{x \in F^{[n] \setminus S} : \Delta_S(c, x) \leq r\} \subseteq F^{[n] \setminus S}$  denote the Hamming ball (with erasures  $S$ ) of radius  $r$  and center  $c$ , and let  $|B_r|$  denote its cardinality (which does not depend on  $c$ ). We have the following basic bounds on  $|B_r|$ :

$$\binom{n - |S|}{r} (|F| - 1)^r \leq |B_r| \leq (r + 1) \binom{n - |S|}{r} (|F| - 1)^r.$$

Since decoding from  $2r$  errors and  $s$  erasures is possible, the Hamming balls  $\{B_{2r}(c)\}_{c \in C}$  are disjoint, and so

$$\begin{aligned} \mathbb{P}_x(\exists c \in C : \Delta_S(c, x) \leq r) &\leq |B_r| / |B_{2r}| \\ &\leq (r + 1) \binom{n - |S|}{r} \binom{n - |S|}{2r}^{-1} (|F| - 1)^{-r} \\ &\leq (r + 1) \binom{n - |S|}{r} \binom{n - |S|}{2r}^{-1}. \end{aligned}$$

Using the standard bounds  $\left(\frac{n}{k}\right)^k \leq \binom{n}{k} \leq \left(\frac{ne}{k}\right)^k$ , this becomes

$$\begin{aligned} &\leq (r + 1) \left( \frac{4er}{n - |S|} \right)^r \\ &\leq (r + 1) \left( \frac{4er}{n - s} \right)^r \end{aligned}$$

which is at most  $(r + 1)2^{-r}$  provided  $r \leq (n - s)/(8e)$ . ◀

► **Lemma 15.** *Let  $j_1, \dots, j_n$  be uniformly and independently chosen from  $[n]$ . For any constant  $\alpha < 1/e$ , the number of indices  $i \in [n]$  that occur exactly once among  $j_1, \dots, j_n$  is at least  $\alpha n$  with high probability.*

**Proof.** Let  $X_i \in \{0, 1\}$  be the indicator that  $i$  occurs exactly once among  $j_1, \dots, j_n$ , and let  $X = \sum_{i=1}^n X_i$ . We have (as  $n \rightarrow \infty$ )

$$\mathbb{E}[X_i] = n(1/n)(1 - 1/n)^{n-1} \rightarrow e^{-1},$$

$$\mathbb{E}[X_i^2] = \mathbb{E}[X_i], \text{ and for } i \neq i',$$

$$\mathbb{E}[X_i X_{i'}] = n(n-1)(1/n)^2(1 - 2/n)^{n-2} \rightarrow e^{-2}.$$

This means  $\mathbb{E}[X] = (1 + o(1))n/e$  and

$$\begin{aligned} \text{Var}(X) &= \sum_i (\mathbb{E}[X_i^2] - \mathbb{E}[X_i]^2) + \sum_{i \neq i'} (\mathbb{E}[X_i X_{i'}] - \mathbb{E}[X_i] \mathbb{E}[X_{i'}]) \\ &\leq n(1 + o(1))(e^{-1} - e^{-2}) + n(n-1) \cdot o(1) \\ &= o(n^2). \end{aligned}$$

The result now follows by Chebyshev's inequality. ◀

## 5.1 Proof of Theorem 4

The idea of the proof is as follows. First imagine the setting where  $\mu$  is not required to be  $S_n$ -invariant. By using each real number to encode an element of  $F = \mathbb{F}_q$ , we can take  $\nu$  to be the uniform distribution on  $F^n$  and take  $\mu$  to be a random Reed-Solomon codeword in  $F^n$ . Under  $T_\delta \mu$ , the noise operator will corrupt a few symbols (“errors”), but the decoding algorithm can correct these and thus distinguish  $T_\delta \mu$  from  $\nu$ .

In order to have  $S_n$ -invariance, we need to modify the construction. Instead of observing an ordered list  $y = (y_1, \dots, y_n)$  of symbols, we will observe pairs of the form  $(i, y_i)$  (with each pair encoded by a single real number) where  $i$  is a random index. If the same  $i$  value appears in two different pairs, this gives conflicting information; we deal with this by simply throwing it out and treating  $y_i$  as an “erasure” that the code needs to correct. If some  $i$  value does not appear in any pairs, we also treat this as an erasure. The full details are given below.

**Proof of Theorem 4.** Let  $C$  be the length- $n$  code from Proposition 11 with  $k = \lceil \alpha n \rceil$  for some constant  $\alpha \in (0, 1)$  to be chosen later. Fix a scheme by which a real number encodes a tuple  $(j, y)$  with  $j \in [n]$  and  $y \in F = \mathbb{F}_q$ , in such a way that a uniformly random real number in  $[0, 1]$  encodes a uniformly random tuple  $(j, y)$ . More concretely, we can take  $n = q = 2^m$  for some integer  $m \geq 1$ , in which case  $(j, y)$  can be directly encoded using the first  $2m$  bits of the binary expansion of a real number. Under  $x \sim \nu$ , each coordinate  $x_i$  encodes an independent uniformly random tuple  $(j_i, y_i)$ . Under  $\mu$ , let each coordinate  $x_i$  be a uniformly random encoding of  $(j_i, y_i)$ , drawn as follows. Let  $\tilde{c}$  be a uniformly random codeword from  $C$ . Draw  $j_1, \dots, j_n \in [n]$  independently and uniformly. For each  $i$ , if  $j_i$  is a unique index (in the sense that  $j_i \neq j_{i'}$  for all  $i' \neq i$ ) then set  $y_i = \tilde{c}_{j_i}$ ; otherwise choose  $y_i$  uniformly from  $F$ .

Note that  $\mu$  is  $S_n$ -invariant. Since the dual distance of  $C$  is  $k + 2 = \Omega(n)$ , it follows (using Proposition 9) that  $\mu$  is  $\Omega(n)$ -wise independent with respect to  $\nu$ . By choosing  $\delta > 0$  and  $\alpha > 0$  sufficiently small, we can ensure that  $16\delta n + (2n/3 + 4\delta n) < n - k$  and so  $C$  admits efficient decoding from  $8\delta n$  errors and  $2n/3 + 4\delta n$  erasures (see Proposition 11). The algorithm to distinguish  $T_\delta \mu$  and  $\nu$  is as follows. Given a list of  $(j_i, y_i)$  pairs, produce  $c' \in F^n$

by setting  $c'_{j_i} = y_i$  wherever  $j_i$  is a unique index (in the above sense), and setting all other positions of  $c'$  to  $\perp$  (an “erasure”). Let  $S \subseteq [n]$  be the indices  $i$  for which  $c'_i = \perp$ . Run the decoding algorithm on  $c'$ ; if it succeeds and outputs a codeword  $c$  such that  $\Delta_S(c, c') \leq 4\delta n$  then output “ $T_\delta \mu$ ”, and otherwise output “ $\nu$ ”.

We can prove correctness as follows. If the true distribution is  $\nu$  then Lemma 15 guarantees  $|S| \leq 2n/3$  (w.h.p.). Since the non-erased values in  $c'$  are uniformly random, Lemma 14 ensures there is no codeword  $c$  with  $\Delta_S(c, c') \leq 4\delta n$  (w.h.p.), provided we choose  $\delta \leq 1/(96e)$ , and so our algorithm outputs “ $\nu$ ” (w.h.p.).

Now suppose the true distribution is  $T_\delta \mu$ . In addition to the  $\leq 2n/3$  erasures caused by non-unique  $j_i$ ’s sampled from  $\mu$ , each coordinate resampled by  $T_\delta$  can create up to 2 additional erasures and can also create up to 2 errors (i.e., coordinates  $i$  for which  $c'_i \neq \perp$  but  $c'_i \neq \tilde{c}_i$ ). Since at most  $2\delta n$  coordinates get resampled (w.h.p.), this means we have a total of (up to)  $4\delta n$  errors and  $2n/3 + 4\delta n$  erasures. This means decoding succeeds and outputs  $\tilde{c}$  (i.e., the true codeword used to sample  $\mu$ ), and furthermore,  $\Delta_S(\tilde{c}, c') \leq 4\delta n$ . ◀

## 5.2 Proof of Theorem 7

The idea of the proof is similar to the previous proof, and somewhat simpler (since we do not need  $S_n$ -invariance). We take  $\nu$  to be the uniform distribution on binary strings and take  $\mu$  to be the uniform distribution on codewords, using the binary code from Proposition 12. The decoding algorithm is able to correct the errors caused by  $T_\delta$ .

**Proof of Theorem 7.** Let  $C$  be the code from Proposition 12. Each codeword  $c \in C$  is an element of  $\mathbb{F}_2^n$ , which can be identified with  $\{0, 1\}^n = \mathcal{X}^n$ . Let  $\mu$  be the uniform distribution over codewords. Since the dual distance of  $C$  is at least  $\zeta n$ , we have from Proposition 9 that  $\mu$  is  $\Omega(n)$ -wise independent with respect to the uniform distribution  $\nu$ . We also know that  $C$  admits efficient decoding from  $\zeta n/2$  errors.

Let  $\delta = \min\{1/(16e), \zeta/8\}$ . The algorithm to distinguish  $T_\delta \mu$  and  $\nu$ , given a sample  $c'$ , is to run the decoding algorithm on  $c'$ ; if decoding succeeds and outputs a codeword  $c$  such that  $\Delta(c, c') \leq 2\delta n$  then output “ $T_\delta \mu$ ”, and otherwise output “ $\nu$ ”.

We can prove correctness as follows. If  $c'$  is drawn from  $T_\delta \mu$  then  $c'$  is separated from some codeword  $c$  by at most  $2\delta n \leq \zeta n/4$  errors (w.h.p.), and so decoding will find  $c$ . If instead  $c'$  is drawn from  $\nu$  then (since  $\delta \leq 1/(16e)$ ) by Lemma 14, there is no codeword within Hamming distance  $2\delta n$  of  $c'$  (w.h.p.). ◀

---

## References

- 1 Afonso S Bandeira, Dmitry Kunisky, and Alexander S Wein. Computational hardness of certifying bounds on constrained PCA problems. *arXiv preprint*, 2019. [arXiv:1902.07324](#).
- 2 Boaz Barak, Chi-Ning Chou, Zhixian Lei, Tselil Schramm, and Yueqi Sheng. (Nearly) efficient algorithms for the graph matching problem on correlated random graphs. In *Advances in Neural Information Processing Systems*, pages 9186–9194, 2019.
- 3 Boaz Barak, Samuel Hopkins, Jonathan Kelner, Pravesh K Kothari, Ankur Moitra, and Aaron Potechin. A nearly tight sum-of-squares lower bound for the planted clique problem. *SIAM Journal on Computing*, 48(2):687–735, 2019.
- 4 Quentin Berthet and Philippe Rigollet. Computational lower bounds for sparse PCA. *arXiv preprint*, 2013. [arXiv:1304.0828](#).
- 5 Avrim Blum, Merrick L. Furst, Jeffrey C. Jackson, Michael J. Kearns, Yishay Mansour, and Steven Rudich. Weakly learning DNF and characterizing statistical query learning using fourier analysis. In Frank Thomson Leighton and Michael T. Goodrich, editors, *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing, 23-25 May 1994, Montréal, Québec, Canada*, pages 253–262. ACM, 1994. [doi:10.1145/195058.195147](#).

- 6 Matthew Brennan and Guy Bresler. Average-case lower bounds for learning sparse mixtures, robust estimation and semirandom adversaries. *arXiv preprint*, 2019. [arXiv:1908.06130](#).
- 7 Yeshwanth Cherapanamjeri, Samuel B Hopkins, Tarun Kathuria, Prasad Raghavendra, and Nilesh Tripuraneni. Algorithms for heavy-tailed statistics: Regression, covariance estimation, and beyond. *arXiv preprint*, 2019. [arXiv:1912.11071](#).
- 8 Yunzi Ding, Dmitriy Kunisky, Alexander S Wein, and Afonso S Bandeira. Subexponential-time algorithms for sparse PCA. *arXiv preprint*, 2019. [arXiv:1907.11635](#).
- 9 David L Donoho, Arian Maleki, and Andrea Montanari. Message-passing algorithms for compressed sensing. *Proceedings of the National Academy of Sciences*, 106(45):18914–18919, 2009.
- 10 G-L Feng and Thammavaranu RN Rao. Decoding algebraic-geometric codes up to the designed minimum distance. *IEEE Transactions on Information Theory*, 39(1):37–45, 1993.
- 11 Venkatesan Guruswami and Madhu Sudan. Improved decoding of reed-solomon and algebraic-geometric codes. In *Proceedings 39th Annual Symposium on Foundations of Computer Science*, pages 28–37. IEEE, 1998.
- 12 Venkatesan Guruswami and Madhu Sudan. On representations of algebraic-geometry codes. *IEEE Transactions on Information Theory*, 47(4):1610–1613, 2001.
- 13 Samuel Hopkins. *Statistical Inference and the Sum of Squares Method*. PhD thesis, Cornell University, 2018.
- 14 Samuel B Hopkins, Pravesh K Kothari, Aaron Potechin, Prasad Raghavendra, Tselil Schramm, and David Steurer. The power of sum-of-squares for detecting hidden structures. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 720–731. IEEE, 2017.
- 15 Samuel B Hopkins and David Steurer. Efficient bayesian estimation from few samples: community detection and related problems. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 379–390. IEEE, 2017.
- 16 Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-SAT. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001. doi:10.1006/jcss.2000.1727.
- 17 Mark Jerrum. Large cliques elude the metropolis process. *Random Structures & Algorithms*, 3(4):347–359, 1992.
- 18 Michael J. Kearns. Efficient noise-tolerant learning from statistical queries. In S. Rao Kosaraju, David S. Johnson, and Alok Aggarwal, editors, *Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing, May 16-18, 1993, San Diego, CA, USA*, pages 392–401. ACM, 1993. doi:10.1145/167088.167200.
- 19 Subhash Khot. On the power of unique 2-prover 1-round games. In *Proceedings of the thirty-fourth annual ACM symposium on Theory of computing*, pages 767–775, 2002.
- 20 Subhash Khot. On the unique games conjecture. In *FOCS*, volume 5, page 3, 2005.
- 21 Dmitriy Kunisky, Alexander S Wein, and Afonso S Bandeira. Notes on computational hardness of hypothesis testing: Predictions using the low-degree likelihood ratio. *arXiv preprint*, 2019. [arXiv:1907.11636](#).
- 22 Thibault Lesieur, Florent Krzakala, and Lenka Zdeborová. MMSE of probabilistic low-rank matrix estimation: Universality with respect to the output channel. In *2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 680–687. IEEE, 2015.
- 23 Sidhanth Mohanty, Prasad Raghavendra, and Jeff Xu. Lifting sum-of-squares lower bounds: Degree-2 to degree-4. *arXiv preprint*, 2019. [arXiv:1911.01411](#).
- 24 Prasad Raghavendra, Tselil Schramm, and David Steurer. High-dimensional estimation via sum-of-squares proofs. *arXiv preprint*, 6, 2018. [arXiv:1807.11419](#).
- 25 Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6):34:1–34:40, 2009. doi:10.1145/1568318.1568324.
- 26 Amir Shpilka. Constructions of low-degree and error-correcting  $\varepsilon$ -biased generators. *computational complexity*, 18(4):495, 2009.
- 27 Alexei N Skorobogatov and Serge G Vladut. On the decoding of algebraic-geometric codes. *IEEE Transactions on Information Theory*, 36(5):1051–1060, 1990.





# High-Entropy Dual Functions and Locally Decodable Codes

Jop Briët

CWI, Science Park 123, 1098 XG Amsterdam, The Netherlands  
j.briet@cwi.nl

Farrokh Labib

CWI, Science Park 123, 1098 XG Amsterdam, The Netherlands  
labib@cwi.nl

---

## Abstract

*Locally decodable codes* (LDCs) allow any single encoded message symbol to be retrieved from a codeword with good probability by reading only a tiny number of codeword symbols, even if the codeword is partially corrupted. LDCs have surprisingly many applications in computer science and mathematics (we refer to [13, 10] for extensive surveys). But despite their ubiquity, they are poorly understood. Of particular interest is the tradeoff between the codeword length  $N$  as a function of message length  $k$  when the *query complexity*—the number of probed codeword symbols—and alphabet size are constant. The Hadamard code is a 2-query LDC of length  $N = 2^{O(k)}$  and this length is optimal in the 2-query regime [11]. For  $q \geq 3$ , near-exponential gaps persist between the best-known upper and lower bounds. The family of Reed-Muller codes, which generalize the Hadamard code, were for a long time the best-known examples, giving  $q$ -query LDCs of length  $\exp(O(k^{1/(q-1)}))$ , until breakthrough constructions of *matching vector* LDCs of Yekhanin and Efremenko [12, 6].

In contrast with other combinatorial objects such as expander graphs, the probabilistic method has so far not been successfully used to beat the best explicit LDC constructions. In [3], a probabilistic framework was given that could in principle yield best-possible LDCs, albeit non-constructively. A special instance of this framework connects LDCs with a probabilistic version of Szemerédi’s theorem. The setup for this is as follows: For a finite abelian group  $G$  of size  $N = |G|$ , let  $D \subseteq G$  be a random subset where each element is present with probability  $\rho$  independently of all others. For  $k \geq 3$  and  $\varepsilon \in (0, 1)$ , let  $E$  be the event that every subset  $A \subseteq G$  of size  $|A| \geq \varepsilon|G|$  contains a proper  $k$ -term arithmetic progression with common difference in  $D$ . For fixed  $\varepsilon > 0$  and sufficiently large  $N$ , it is an open problem to determine the smallest value of  $\rho$ —denoted  $\rho_k$ —such that  $\Pr[E] \geq \frac{1}{2}$ . In [3] it is shown that there exist  $k$ -query LDCs of message length  $\Omega(\rho_k N)$  and codeword length  $O(N)$ . As such, Szemerédi’s theorem with random differences, in particular lower bounds on  $\rho_k$ , can be used to show the existence of LDCs. Conversely, this connection indirectly implies the best-known upper bounds on  $\rho_k$  for all  $k \geq 3$  [8, 4]. However, a conjecture from [9] states that over  $\mathbb{Z}_N$  we have  $\rho_k \leq O_k(N^{-1} \log N)$  for all  $k$ , which would be best-possible. Truth of this conjecture would imply that over this group, Szemerédi’s theorem with random differences cannot give LDCs better than the Hadamard code. For finite fields, Altman [1] showed that this is false. In particular, over  $\mathbb{F}_p^n$  for  $p$  odd, he proved that  $\rho_3 \geq \Omega(p^{-n} n^2)$ ; generally,  $\rho_k \geq \Omega(p^{-n} n^{k-1})$  holds when  $p \geq k + 1$  [2]. In turn, these bounds are conjectured to be optimal for the finite-field setting, which would imply that over finite fields, Szemerédi’s theorem with random differences cannot give LDCs better than Reed-Muller codes.

The finite-field conjecture is motivated mainly by the possibility that so-called *dual functions* can be approximated well by *polynomial phases*, functions of the form  $e^{2\pi i P(x)/p}$  where  $P$  is a multivariate polynomial over  $\mathbb{F}_p$ . We show that this is false. Using Yekhanin’s matching-vector-code construction, we give dual functions of order  $k$  over  $\mathbb{F}_p^n$  that cannot be approximated in  $L_\infty$ -distance by polynomial phases of degree  $k - 1$ . This answers in the negative a natural finite-field analog of a problem of Frantzikinakis over  $\mathbb{N}$  [7, Problem 1].

**2012 ACM Subject Classification** Theory of computation

**Keywords and phrases** Higher-order Fourier analysis, dual functions, finite fields, additive combinatorics, coding theory



© Jop Briët and Farrokh Labib;

licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 76; pp. 76:1–76:2

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Digital Object Identifier 10.4230/LIPIcs.ITCS.2021.76

Category Extended Abstract

Related Version This is an extended abstract of [5], <https://arxiv.org/abs/2010.14956>.

Funding Jop Briët: Supported by the Gravitation grant NETWORKS-024.002.003 from the Dutch Research Council (NWO)

Farrokh Labib: Supported by the Gravitation grant NETWORKS-024.002.003 from the Dutch Research Council (NWO)

---

## References

- 1 Daniel Altman. On Szemerédi’s theorem with differences from a random set. *Acta Arith.*, 195:97–108, 2020. doi:10.4064/aa190531-25-10.
- 2 Jop Briët. Subspaces of tensors with high analytic rank. *Online Journal of Analytic Combinatorics*, 2020. To appear. Available at arXiv: 1908.04169.
- 3 Jop Briët, Zeev Dvir, and Sivakanth Gopi. Outlaw distributions and locally decodable codes. *Theory of Computing*, 15(12):1–24, 2019. Preliminary version in ITCS’17. doi:10.4086/toc.2019.v015a012.
- 4 Jop Briët and Sivakanth Gopi. Gaussian width bounds with applications to arithmetic progressions in random settings. *International Mathematics Research Notices*, page rny238, 2018. doi:10.1093/imrn/rny238.
- 5 Jop Briët and Farrokh Labib. High-entropy dual functions over finite fields and locally decodable codes. *arXiv preprint*, 2020. arXiv:2010.14956.
- 6 Klim Efremenko. 3-Query locally decodable codes of subexponential length. *SIAM J. Comput.*, 41(6):1694–1703, 2012. Preliminary version in STOC’09. doi:10.1137/090772721.
- 7 Nikos Frantzikinakis. Some open problems on multiple ergodic averages. *Bull. Hellenic Math. Soc.*, 60:41–90, 2016. doi:10.1109/tac.2015.2392613.
- 8 Nikos Frantzikinakis, Emmanuel Lesigne, and Mate Wierdl. Random sequences and pointwise convergence of multiple ergodic averages. *Indiana Univ. Math. J.*, pages 585–617, 2012.
- 9 Nikos Frantzikinakis, Emmanuel Lesigne, and Mate Wierdl. Random differences in Szemerédi’s theorem and related results. *J. Anal. Math.*, 130:91–133, 2016. doi:10.1007/s11854-016-0030-z.
- 10 Sivakanth Gopi. *Locality in coding theory*. PhD thesis, Princeton University, 2018.
- 11 Iordanis Kerenidis and Ronald de Wolf. Exponential lower bound for 2-query locally decodable codes via a quantum argument. *J. Comput. System Sci.*, 69(3):395–420, 2004. Earlier version in STOC’03. doi:10.1016/j.jcss.2004.04.007.
- 12 Sergey Yekhanin. Towards 3-query locally decodable codes of subexponential length. *J. ACM*, 55(1):1:1–1:16, 2008. Preliminary version in STOC’07. doi:10.1145/1326554.1326555.
- 13 Sergey Yekhanin. Locally decodable codes. *Found. Trends Theor. Comput. Sci.*, 6(3):139–255, 2012. doi:10.1561/04000000030.

# Buying Data over Time: Approximately Optimal Strategies for Dynamic Data-Driven Decisions

Nicole Immorlica

Microsoft Research, Cambridge, MA, USA  
nicimm@microsoft.com

Ian A. Kash

Department of Computer Science, University of Illinois at Chicago, IL, USA  
iankash@uic.edu

Brendan Lucier

Microsoft Research, Cambridge, MA, USA  
brlucier@microsoft.com

---

## Abstract

We consider a model where an agent has a repeated decision to make and wishes to maximize their total payoff. Payoffs are influenced by an action taken by the agent, but also an unknown state of the world that evolves over time. Before choosing an action each round, the agent can purchase noisy samples about the state of the world. The agent has a budget to spend on these samples, and has flexibility in deciding how to spread that budget across rounds. We investigate the problem of choosing a sampling algorithm that optimizes total expected payoff. For example: is it better to buy samples steadily over time, or to buy samples in batches? We solve for the optimal policy, and show that it is a natural instantiation of the latter. Under a more general model that includes per-round fixed costs, we prove that a variation on this batching policy is a 2-approximation.

**2012 ACM Subject Classification** Theory of computation → Design and analysis of algorithms; Theory of computation → Markov decision processes

**Keywords and phrases** Online Algorithms, Value of Data, Markov Processes

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.77

**Related Version** The full version of this paper is available at <https://arxiv.org/abs/2101.07304>.

**Acknowledgements** Part of this work was done while Ian Kash was at Microsoft Research. He gratefully acknowledges support from the National Science Foundation via award CCF 1934915.

## 1 Introduction

The growing demand for machine learning practitioners is a testament to the way data-driven decision making is shaping our economy. Data has proven so important and valuable because so much about the current state of the world is *a priori* unknown. We can better understand the world by investing in data collection, but this investment can be costly; deciding how much data to acquire can be a non-trivial undertaking, especially in the face of budget constraints. Furthermore, the value of data is typically not linear. Machine learning algorithms often see diminishing returns to performance as their training dataset grows [22, 10]. This non-linearity is further complicated by the fact that a data-driven decision approach is typically intended to replace some existing method, so its value is relative to the prior method's performance.

As a motivating example for these issues, consider a politician who wishes to accurately represent the opinion of her constituents. These constituents have a position on a policy, say the allocation of funding to public parks. The politician must choose her own position on the policy or abstain from the discussion. If she states a position, she experiences a disutility that is increasing in the distance of her position from that of her constituents. If she abstains, she incurs a fixed cost for failing to take a stance. To help her make an optimal decision she can hire a polling firm that collects data on the participants' positions.



© Nicole Immorlica, Ian A. Kash, and Brendan Lucier;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 77; pp. 77:1–77:14



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

We focus on the dynamic element of this story. In many decision problems, the state of the world evolves over time. In the example above, the opinions of the constituents might change as time passes, impacting the optimal position of the politician. As a result, data about the state of the world becomes stale. Furthermore, many decisions are not made a single time; instead, decisions are made repeatedly. In our example, the politician can update funding levels each fiscal quarter.

When faced with budget constraints on data collection and the issue of data staleness, decisions need to be made about when to collect data and when to save budget for the future, and whether to make decisions based on stale data or apply a default, non-data-driven policy. Our main contribution is a framework that models the impact of such budget constraints on data collection strategies. In our example, the politician has a budget for data collection. A polling firm charges a fixed cost to initiate a poll (e.g., create the survey) plus a fee per surveyed participant. The politician may not have enough budget to hire the firm to survey every constituent every quarter. Should she then survey fewer constituents every quarter? Or survey a larger number of constituents every other quarter, counting on the fact that opinions do not drift too rapidly?

We initiate the study with arguably the simplest model that exhibits this tension. The state of the world (constituents' opinions) is hidden but drawn from a known prior distribution, then evolves stochastically. Each round, the decision-maker (politician) can collect one or more noisy samples that are correlated with the hidden state at a cost affine in the number of samples (conduct a poll). Then she chooses an action and incurs a loss. Should the decision-maker not exhaust her budget in a given round, she can bank it for future rounds. A sampling algorithm describes an online policy for scheduling the collection of samples given the budget and past observations.

We instantiate this general framework by assuming Gaussian prior, perturbations and sample noise.<sup>1</sup> We capture the decisions that need to be made as the problem of estimating the current state value, using the classic squared loss to capture the cost of making a decision using imprecise information. Alternatively, there is always the option to not make a decision based on the data and instead accept a default constant loss. We assume a budget on the number of samples collected per unit time, and importantly this budget can be banked for future rounds if desired.

## 1.1 A Simple Example

To illustrate our technical model, suppose the hidden state (constituents' average opinion) is initially drawn from a mean-zero Gaussian of variance 1. In each round, the state is subject to mean-zero Gaussian noise of variance 1 (the constituents update their opinions), which is added to the previous round's state. Also, any samples we choose to take are also subject to mean-zero Gaussian noise of variance 1 (polls are imperfect). Our budget for samples is 1 per period, and one can either guess at the hidden state (incurring a penalty equal to the squared loss) or pass and take a default loss of  $3/4$ . What is the expected average loss of the policy that takes a single sample each round, and then takes the optimal action? As it turns out, the expected loss is precisely  $\phi - 1 \approx 0.618$ , where  $\phi$  is the golden ratio  $\frac{1+\sqrt{5}}{2}$  (see Section 3.5 for the analysis). However, this is not optimal: saving up the allotted budget and taking

---

<sup>1</sup> A Gaussian prior is justified in our running example if we assume a large population limit of constituents' opinions. That the prior estimate of drift is also Gaussian is likewise motivated as the number of periods grows large. We discuss alternative distributional assumptions on the prior, perturbations and noise in Section 6.

two samples every other round leads to an expected loss of  $\frac{0.75+\sqrt{2}-1}{2} \approx 0.582$ . The intuition behind the improvement is that taking a single sample every round beats the outside option, but not by much; it is better to beat the outside option significantly on even-numbered rounds (by taking 2 samples), then simply use the outside option on odd-numbered rounds. It turns out that one cannot improve on this by saving up for 3 or more rounds to take even more samples all at once. However, one can do better by alternating between taking no samples for two periods and then two samples each for two periods, which results in a long-run average loss of  $\approx 0.576$ .

## 1.2 Our Results

As we can see from the example above, the space of policies to consider is quite large. One simple observation is that since samples become stale over time it is never optimal to collect samples and then take the outside option (i.e., default fixed-cost action) in the same round; it would be better to defer data collection to later rounds where decisions will be made based on data. As a result, a natural class of policies to consider is those which alternate between collecting samples and saving budget. Such “on-off” policies can be thought of as engaging in “data drives” while neglecting data collection the rest of the time.

Our main result is that these on-off policies are asymptotically optimal, with respect to all dynamic policies. Moreover, it suffices to collect samples at a constant rate during the sampling part of the policy’s period. Our argument is constructive, and we show how to compute an asymptotically optimal policy. This policy divides time into exponentially-growing chunks and collects data in the latter end of each chunk.

The solution above assumes that costs are linear in the number of samples collected. We next consider a more general model with a fixed up-front cost for the first sample collected in each round. This captures the costs associated with setting up the infrastructure to collect samples on a given round, such as hiring a polling firm which uses a two-part tariff. Under such per-round costs, it can be suboptimal to sample in sequential periods (as in an on-off policy), as this requires paying the fixed cost twice. For this generalized cost model, we consider simple and approximately optimal policies. When evaluating performance, we compare against a null “baseline” policy that eschews data collection and simply takes the outside option every period. We define the value of a policy to be its improvement over this baseline, so that the null policy has a value of 0 and every policy has non-negative value. While this is equivalent to simply comparing the expected costs of policies this alternative measure is intended to capture how well a policy leverages the extra value obtainable from data; we feel that this more accurately reflects the relative performance of different policies.

We focus on a class of *lazy policies* that collect samples only at times when the variance of the current estimate is worse than the outside option. This class captures a heuristic based on a threshold rule: the decision-maker chooses to collect data when they do not have enough information to gain over the outside option. We show the optimal lazy policy is a  $1/2$ -approximation to the optimal policy. The result is constructive, and we show how to compute an asymptotically optimal lazy policy. Moreover, this approximation factor is tight for lazy policies.

To derive these results, we begin with the well-known fact that the expected loss under the squared loss cost function is the variance of the posterior. We use an analysis based on Kalman filters [23], which are used to solve localization problems in domains such as astronautics [27], robotics [34], and traffic monitoring [36], to characterize the evolution of variance given a sampling policy. We show how to maximize value using geometric arguments and local manipulations to transform an optimal policy into either an on-off policy or a lazy policy, respectively.

We conclude with two extensions. We described our results for a discrete-time model, but one might instead consider a continuous-time variant in which samples, actions, and state evolution occur continuously. We show how to extend all of our results to such a continuous setting. Second, we describe a non-Gaussian instance of our framework, where the state of the world is binary and switches with some small probability each round. We solve for the optimal policy, and show that (like the Gaussian model) it is characterized by non-uniform, bursty sampling.

### 1.3 Other Motivating Examples

We motivated our framework with a toy example of a politician polling his or her constituents. But we note that the model is general and applies to other scenarios as well. For example, suppose a phone uses its GPS to collect samples, each of which provides a noisy estimate of location (reasonably approximated by Gaussian noise). The “cost” of collecting samples is energy consumption, and the budget constraint is that the GPS can only reasonably use a limited portion of the phone’s battery capacity. The worse the location estimate is, the less useful this information is to apps; sufficiently poor estimates might even have negative value. However, as an alternative, apps always have the outside option of providing location-unaware functionality. Our analysis shows that it is approximately optimal to extrapolate from existing data to estimate the user’s location most of the time, and only use the GPS in “bursts” once the noise of the estimate exceeds a certain threshold. Note that in this scenario the app never observes the “ground truth” of the phone’s location. Similarly, our model might capture the problem faced by a firm that runs user studies when deciding which features to include in a product, given that such user studies are expensive to run and preferences may shift within the population of customers over time.

### 1.4 Future Directions

Our results provide insight into the trade-offs involved in designing data collection policies in dynamic settings. We construct policies that navigate the trade-off between cost of data collection and freshness of data, and show how to optimize data collection schedules in a setting with Gaussian noise. But perhaps our biggest contribution is conceptual, in providing a framework in which these questions can be formalized and studied. We view this work as a first step toward a broader study of the dynamic value of data. An important direction for future work is to consider other models of state evolution and/or sampling within our framework, aimed at capturing other applications. For example, if the state evolves in a heavy-tailed manner, as in the non-Gaussian instance explored in Section 6, then we show it is beneficial to take samples regularly in order to detect large, infrequent jumps in state value, and then adaptively take many samples when such a jump is evident. We solve this extension only for a simple two-state Markov chain. Can we quantify the dynamic value of data and find an (approximately) optimal and simple data collection policy in a general Markov chain?

### 1.5 Related work

While we are not aware of other work addressing the value of data in a dynamic setting, there has been considerable attention paid to the value of data in static settings. Arietta-Ibarra et al. [4] argue that the data produced by internet users is so valuable that they should be compensated for their labor. Similarly, there is growing appreciation for the value of the

data produced on crowdsourcing platforms like Amazon Mechanical Turk [6, 20]. Other work has emphasized that not all crowdsourced data is created equal and studied the way tasks and incentives can be designed to improve the quality of information gathered [17, 30]. Similarly, data can have non-linear value if individual pieces are substitutes or complements [8]. Prediction markets can be used to gather information over time, with participants controlling the order in which information is revealed [11].

There is a growing line of work attempting to determine the marginal value of training data for deep learning methods. Examples include training data for classifying medical images [9] and chemical processes [5], as well as for more general problems such as estimating a Gaussian distribution [22]. These studies consider the static problem of learning from samples, and generally find that additional training data exhibits decreasing marginal value. Koh and Liang [25] introduced the use of influence functions to quantify how the performance of a model depends on individual training examples.

While we assume samples are of uniform quality, other work has studied agents who have data of different quality or cost [29, 7, 16]. Another line studies the way that data is sold in current marketplaces [32], as well as proposing new market designs [28]. This includes going beyond markets for raw data to markets which acquire and combine the outputs of machine learning models [33].

Our work is also related to statistical and algorithmic aspects of learning a distribution from samples. A significant body of recent work has considered problems of learning Gaussians using a minimal number of noisy and/or adversarial samples [21, 13, 14, 26, 15]. In comparison, we are likewise interested in learning a hidden Gaussian from which we obtain noisy samples (as a step toward determining an optimal action), but instead of robustness to adversarial noise we are instead concerned about optimizing the split of samples across time periods in a purely stochastic setting.

Our investigation of data staleness is closely related to the issue of concept drift in streaming algorithms; see, e.g., Chapter 3 of [2]. Concept drift refers to scenarios where the data being fed to an algorithm is pulled from a model that evolves over time, so that, for example, a solution built using historical data will eventually lose accuracy. Such scenarios arise in problems of histogram maintenance [18], dynamic clustering [3], and others. One problem is to quantify the amount of drift occurring in a given data stream [1]. Given that such drift is present, one approach to handling concept drift is via sliding-window methods, which limit dependence on old data [12]. The choice of window size captures a tension between using a lot of stale data or a smaller amount of fresh data. However, in work on concept drift one typically cannot control the rate at which data is collected.

Another concept related to staleness is the “age of information.” This captures scenarios where a source generates frequent updates and a receiver wishes to keep track of the current state, but due to congestion in the transmission technology (such as a queue or database locks) it is optimal to limit the rate at which updates are sent [24, 31]. Minimizing the age of information can be captured as a limit of our model where a single sample suffices to provide perfect information. Recent work has examined variants of the model where generating updates is costly [19], but the focus in this literature is more on the management of the congestible resource. Closer to our work, several recent papers have eliminated the congestible resource and studied issues such as an energy budget that is stochastic and has limited storage capacity [37] and pricing schemes for when sampling costs are non-uniform [35, 38]. Relative to our work these papers have simpler models of the value of data and focus on features of the sampling policy given the energy technology and pricing scheme, respectively.



## 2 Model

We first describe our general framework, then describe a specific instantiation of interest in Section 2.1. Time occurs in rounds, indexed by  $t = 1, 2, \dots$ . There is a hidden state variable  $x_t \in \Omega$  that evolves over time according to a stochastic process. The initial state  $x_1$  is drawn from known distribution  $F_1$ . Write  $m_t$  for the (possibly randomized) evolution mapping applied at round  $t$ , so that  $x_{t+1} \leftarrow m_t(x_t)$ .

In every round, the decision-maker chooses an action  $y_t \in A$ , and then suffers a loss  $\ell(y_t, x_t)$  that depends on both the action and the hidden state. The evolution functions ( $m_t$ ) and loss function  $\ell$  are known to the decision-maker, but neither the state  $x_t$  nor the loss  $\ell(y_t, x_t)$  is directly observed.<sup>2</sup> Rather, on each round before choosing an action, the decision-maker can request one or more independent samples that are correlated with  $x_t$ , drawn from a known distribution  $\Gamma(x_t)$ .

Samples are costly, and the decision-maker has a budget that can be used to obtain samples. The budget is  $B$  per round, and can be banked across rounds. A sampling policy results in a number of samples  $s_t$  taken in each round  $t$ , which can depend on all previous observations. The cost of taking  $s_t$  samples in round  $t$  is  $C(s_t) \geq 0$ . We assume that  $C$  is non-decreasing and  $C(0) = 0$ . A sampling policy is *valid* if  $\sum_{t=1}^T C(s_t) \leq B \cdot T$  for all  $T$ . For example,  $C(s_t) = s_t$  corresponds to a cost of 1 per sample, and setting  $C(s_t) = s_t + z \cdot \mathbb{1}_{s_t > 0}$  adds an additional cost of  $z$  for each round in which at least one sample is collected.

To summarize: on each round, the decision-maker chooses a number of samples  $s_t$  to observe, then chooses an action  $y_t$ . Their loss  $\ell(y_t, x_t)$  is then realized, the value of  $x_t$  is updated to  $x_{t+1}$ , and the process proceeds with the next round. The goal is to minimize the expected long-run average of  $\ell(y_t, x_t)$ , in the limit as  $t \rightarrow \infty$ , subject to  $\sum_{t=0}^T C(s_t) \leq B \cdot T$  for all  $T \geq 1$ .

### 2.1 Estimation under Gaussian Drift

We will be primarily interested in the following instantiation of our general framework. The hidden state variable is a real number (i.e.,  $\Omega = \mathbb{R}$ ) and the decision-maker's goal is to estimate the hidden state in each round. The initial state is  $x_1 \sim N(0, \rho)$ , a Gaussian with mean 0 and variance  $\rho > 0$ . Moreover, the evolution process  $m_t$  sets  $x_{t+1} = x_t + \delta_t$ , where each  $\delta_t \sim N(0, \rho)$  independently. We recall that the decision-maker knows the evolution process (and hence  $\rho$ ) but does not directly observe the realizations  $\delta_t$ .

Each sample in round  $t$  is drawn from  $N(x_t, \sigma)$  where  $\sigma > 0$ . Some of our results will also allow fractional sampling, where we think of an  $\alpha \in (0, 1)$  fraction of a sample as a sample drawn from  $N(x_t, \sigma/\alpha)$ .<sup>3</sup> The action space is  $A = \mathbb{R} \cup \{\perp\}$ . If the decision-maker chooses  $y_t \in \mathbb{R}$ , her loss is the squared error of her estimate  $(y_t - x_t)^2$ . If she is too unsure of the state, she may instead take a default action  $y_t = \perp$ , which corresponds to not making a guess; this results in a constant loss of  $c > 0$ . Let  $G_t$  be a random variable whose law

<sup>2</sup> Assuming that the ground truth for  $\ell(y_t, x_t)$  is unobserved captures scenarios like our political example, and approximates settings where the decision maker only gets weak feedback, feedback at a delay, or feedback in aggregate over a long period of time. Observing the loss provides additional information about  $x_{t+1}$ , and this could be considered a variant of our model where the decision-maker gets some number of samples “for free” each round from observing a noisy version of the loss.

<sup>3</sup> One can view fractional sampling as modeling scenarios where the value of any one single sample is quite small; i.e., has high variance, so that a single “unit” of variance is derived from taking many samples. E.g., sampling a single constituent in our polling example. It also captures settings where it is possible to obtain samples of varying quality with different levels of investment.

is the decision maker's posterior after observing whatever samples are taken in round  $t$  as well as all previous samples. The decision maker's subjective expected loss when guessing  $y_t \in \mathbb{R}$  is  $E[(y_t - G_t)^2]$ . This is well known to be minimized by taking  $y_t = E[G_t]$ , and that furthermore the expected loss is  $E[(E[G_t] - G_t)^2] = \text{Var}(G_t)$ . It is therefore optimal to guess  $y_t = E[G_t]$  if and only if  $\text{Var}(G_t) < c$ , otherwise pass.

We focus on deriving approximately optimal sampling algorithms. To do so, we need to track the variance of  $G_t$  as a function of the sampling strategy. As the sample noise and random state permutations are all zero-mean Gaussians,  $G_t$  is a zero-mean Gaussian as well, and the evolution of its variance has a simple form.

► **Lemma 1.** *Let  $v_t$  be the variance of  $G_t$  and suppose each  $\delta_t \sim N(0, \rho)$  independently, and that each sample is subject to zero-mean Gaussian noise with variance  $\sigma$ . Then, if the decision-maker takes  $s$  samples in round  $t + 1$ , the variance of  $G_{t+1}$  is*

$$v_{t+1} = \frac{v_t + \rho}{1 + \frac{s}{\sigma}(v_t + \rho)}.$$

The proof, which is deferred to the full version of the paper along with all other proofs, follows from our model being a special case of the model underlying a Kalman filter.

The optimization problem therefore reduces to choosing a number of samples  $s_t$  to take in each round  $t$  in order to minimize the long-run average of  $\min(v_t, c)$ , the loss of the optimal action. That is, the goal is to minimize  $\limsup_{T \rightarrow \infty} \frac{1}{T} \sum_{t=1}^T \min(v_t, c)$ , where we take the superior limit so that the quantity is defined even when the average is not convergent. We choose  $C(s_t) = s_t + z \cdot \mathbb{1}_{s_t > 0}$ , so this optimization is subject to the budget constraint that, at each time  $T \geq 1$ ,  $\sum_{t=1}^T s_t + z \cdot \mathbb{1}_{s_t > 0} \leq BT$ . This captures two kinds of information acquisition costs faced by the decision-maker. First she faces a cost per sample, which we have normalized to one. Second, she faces a fixed cost  $z$  (which may be 0) on each day she chooses to take samples, expressed in terms of the number of samples that could instead have been taken on some other day had this cost not been paid. This captures the costs associated with setting up the infrastructure to collect samples on a given round, such as getting data collectors to the location where they are needed, hiring a polling firm which uses a two-part tariff, or establishing a satellite connection to begin using a phone's GPS.

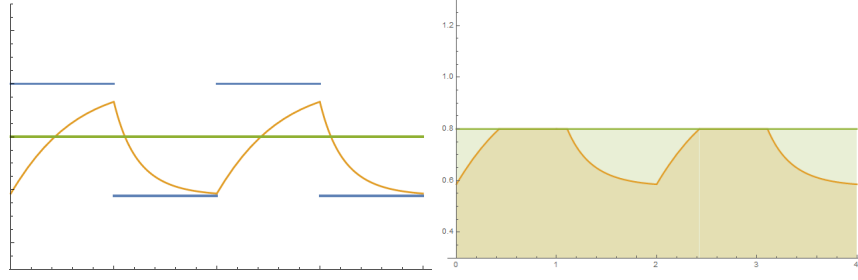
A useful baseline performance is the cost of a policy that takes no samples and simply chooses the outside option at all times. We refer to this as the *null policy*. The *value* of a sampling policy  $s$ , denoted  $\text{Val}(s)$ , is defined to be the difference between its cost and the cost of the null policy:  $\liminf_{T \rightarrow \infty} \frac{1}{T} \sum_{t=1}^T \max(c - v_t, 0)$ . Note that maximizing value is equivalent to minimizing cost, which we illustrate in Section 3.1. We say that a policy is  $\alpha$ -approximate if its value is at least an  $\alpha$  fraction of the optimal policy's value.

### 3 Analyzing Variance Evolution

Before moving on to our main results, we show how to analyze the evolution of the variance resulting from a given sampling policy. We first illustrate our model with a particularly simple class of policies: those where  $s_t$  takes on only two possible values. We then analyze arbitrary periodic policies, and show via contraction that they result in convergence to a periodic variance evolution.

#### 3.1 Visualizing the Decision Problem

To visualize the problem, we begin by plotting the result of an example policy where the spending rate is constant for some interval of rounds, then shifts to a different constant spending rate. Figure 1 illustrates one such policy. The spending rates are indicated as



■ **Figure 1** The variance for a piecewise-constant sampling policy, and its loss and benefit.

alternating line segments, while the variance is an oscillating curve, always converging toward the current spending rate. Note that this particular policy is periodic, in the sense that the final variance is the same as the initial variance. The horizontal line gives one possible value for the cost of the outside option. Given this, the optimal policy is to guess whenever the orange curve is below the green line and take the outside option whenever it is above it. Thus, the loss associated with this spending policy is given by the orange shaded area in Figure 1. Minimizing this loss is equivalent to maximizing the green shaded area, which corresponds to the value of the spending policy. The null policy, which takes no samples and has variance greater than  $c$  always (possibly after an initial period if  $v_0 < c$ ), has value 0.

### 3.2 Periodic Policies

We next consider policies that are periodic. A *periodic policy* with period  $R$  has the property that  $s_t = s_{t+R}$  for all  $t \geq 1$ . Such policies are natural and have useful structure. In a periodic policy, the variance ( $v_t$ ) converges uniformly to being periodic in the limit as  $t \rightarrow \infty$ . This follows because the impact of sampling on variance is a contraction map.

► **Definition 2.** Given a normed space  $X$  with norm  $\|\cdot\|$ , a mapping  $\Psi: X \rightarrow X$  is a contraction map if there exists a  $k < 1$  such that, for all  $x, y \in X$ ,  $\|\Psi(x) - \Psi(y)\| \leq k\|x - y\|$ .

► **Lemma 3.** Fix a sampling policy  $s$ , and a time  $R \geq 1$ , and suppose that  $s$  takes a strictly positive number of samples in each round  $t \leq R$ . Let  $\Psi$  be the mapping defined as follows: supposing that  $v_0 = x$  and  $v$  is the variance function resulting from sampling policy  $s$ , set  $\Psi(x) := v_R$ . Then  $\Psi$  is a contraction map over the non-negative reals, under the absolute value norm.

The proof appears in the full version of the paper. It is well known that a contraction map has a unique fixed point, and repeated application will converge to that fixed point. Since we can view the impact of the periodic sampling policy as repeated application of mapping  $\Psi$  to the initial variance in order to obtain  $v_0, v_R, v_{2R}, \dots$ , we conclude that the variance will converge uniformly to a periodic function for which  $v_t = v_{t+R}$ . Thus, for the purpose of evaluating long-run average cost, it will be convenient (and equivalent) to replace the initial condition on  $v_0$  with a periodic boundary condition  $v_0 = v_R$ , and then choose  $s$  to minimize the average cost over a single period,  $\frac{1}{R} \int_0^R \min\{v_t, c\} dt$ , subject to the budget constraint that, at any round  $T \leq R$ , we have  $\sum_{t=1}^T s_t \leq BT$ .

### 3.3 Lazy Policies

Write  $\tilde{v} = v_{t-1} + \rho$  for the variance that would be obtained in round  $t$  if  $s_t = 0$ . We say that a policy is *lazy* if  $s_t = 0$  whenever  $\tilde{v}_t < c$ . That is, samples are collected only at times where the variance would otherwise be at or above the outside option value  $c$ . Intuitively, we can

think of such a policy as collecting a batch of samples in one round, then “free-riding” off of the resulting information in subsequent rounds. The free-riding occurs until the posterior variance grows large enough that it becomes better to select the outside option, at which point the policy may collect another batch of samples.

If a policy is lazy, then its variance function  $v$  increases by  $\rho$  whenever  $\tilde{v}_t < c$ , with downward steps only at times corresponding to when samples are taken. Furthermore, the value of such a policy decomposes among these sampling instances: for any  $t$  where  $s_t > 0$ , resulting in a variance of  $v_t < c$ , if we write  $h = \lfloor c - v_t \rfloor$  then we can attribute a value of  $\frac{1}{2}h(h+1) + (h+1)(c - v_t - h)$ . Geometrically, this is the area of the “discrete-step triangle” formed between the increasing sequence of variances  $v_t$  and the constant line at  $c$ , over the time steps  $t, \dots, t+h+1$ .

### 3.4 On-Off Policies

An On-Off policy is a periodic policy parameterized by a time interval  $T$  and a sampling rate  $S$ . Roughly speaking, the policy alternates between intervals where it samples at a rate of  $S$  each round, and intervals where it does not sample. The two interval lengths sum to  $T$ , and the length of the sampling interval is set as large as possible subject to budget feasibility. More formally, the policy sets  $s_t = 0$  for all  $t \leq (1 - \alpha) \cdot T$ , where  $\alpha = \min\{B/S, 1\} \in [0, 1]$  and  $s_t = S$  for all  $t$  such that  $(1 - \alpha)T < t \leq T$ . This policy is then repeated, on a cycle of length  $T$ . The fraction  $\alpha$  is chosen to be as large as possible, subject to the budget constraint.

### 3.5 Simple Example Revisited

We can now justify the simple example we presented in the introduction, where  $\rho = \sigma = 1$ ,  $B = 1$ , and  $c = 0.75$ . The policy that takes a single sample each round is periodic with period 1, and hence will converge to a variance that is likewise equal each round. This fixed point variance,  $v^*$ , satisfies  $v^* = \frac{v^*+1}{1+(v^*+1)}$  by Lemma 1. Solving for  $v^*$  yields  $v^* = \frac{\sqrt{5}-1}{2} < 0.75$ , which is the average cost per round.

If instead the policy takes  $k$  samples every  $k$  rounds, this results in a variance that is periodic of period  $k$ . After the round in which samples are taken, the fixed-point variance satisfies  $v^* = \frac{v^*+k}{1+k(v^*+k)}$ , again by Lemma 1. Solving for  $v^*$ , and noting that  $v^* + 1 \geq 1 > c$ , yields that the cost incurred by this policy is minimized when  $k = 2$ .

To solve for the policy that alternates between taking no samples for two round, followed by taking two samples on each of two rounds, suppose the long-run, periodic variances are  $v_1, v_2, v_3, v_4$ , where samples are taken on rounds 3 and 4. Then we have  $v_2 = v_1 + 1$ ,  $v_3 = \frac{v_2+1}{1+2(v_2+1)}$ ,  $v_4 = \frac{v_3+1}{1+2(v_3+1)}$ , and  $v_1 = v_4 + 1$ . Combining this sequence of equations yields  $4v_1^2 + 4v_1 - 13 = 0$ , which we can solve to find  $v_1 = \frac{-1+\sqrt{14}}{2} \approx 1.3708$ . Plugging this into the equations for  $v_2, v_3, v_4$  and taking the average of  $\min\{v_i, 0.75\}$  over  $i \in \{1, 2, 3, 4\}$  yields the reported average cost of  $\approx 0.576$ .

## 4 Solving for the Optimal Policy

In this section we show that when the cost of sampling is linear in the total number of samples taken (i.e.,  $z = 0$ )<sup>4</sup>, and when fractional sampling is allowed, then the supremum value over all on-off policies is an upper bound on the value of any policy. This supremum

<sup>4</sup> Recall that  $z$  is the fixed per-round cost of taking a positive number of samples. Even when  $z = 0$ , there is still a positive per-sample cost.

is achieved in the limit as the time interval  $T$  grows large. So, while no individual policy achieves the supremum, one can get arbitrarily close with an on-off policy of sufficiently long period. Proofs appear in the full version of the paper.

We begin with some definitions. For a given period length  $T > 0$ , write  $s^T$  for the on-off policy of period  $T$  with optimal long-run average value. Recall  $\text{Val}(s^T)$  is the value of policy  $s^T$ . We first argue that larger time horizons lead to better on-off policies.

► **Lemma 4.** *With fractional samples, for all  $T > T'$ , we have  $\text{Val}(s^T) > \text{Val}(s^{T'})$ .*

Write  $V^* = \sup_{T \rightarrow \infty} \text{Val}(s^T)$ . Lemma 4 implies that  $V^* = \lim_{T \rightarrow \infty} \text{Val}(s^T)$  as well. We show that no policy satisfying the budget constraint can achieve value greater than  $V^*$ .

► **Theorem 5.** *With fractional samples, the value of any valid policy  $s$  is at most  $V^*$ .*

The proof of Theorem 5 proceeds in two steps. First, for any given time horizon  $T$ , it is suboptimal to move from having variance below the outside option to above the outside option; one should always save up budget over the initial rounds, then keep the variance below  $c$  from that point onward. This follows because the marginal sample cost of reducing variance diminishes as variance grows, so it is more sample-efficient to recover from very high variance once than to recover from moderately high variance multiple times.

Second, one must show that it is asymptotically optimal to keep the variance not just below  $c$ , but uniform. This is done by a potential argument, illustrating that a sequence of moves aimed at “smoothing out” the sampling rate can only increase value and must terminate at a uniform policy. The difficulty is that a sample affects not only the value in the round it is taken, but in all subsequent rounds. We make use of an amortization argument that appropriately credits value to samples, and use this to construct the sequence of adjustments that increase overall value while bringing the sampling sequence closer to uniform in an appropriate metric.

We also note that it is straightforward to compute the optimal on-off policy for a given time horizon  $T$ , by choosing the sampling rate that maximizes [value per round]  $\times$  [fraction of time the policy is “on”]. One can implement a policy whose value asymptotically approaches  $V^*$  by repeated doubling of the time horizon. Alternatively, since  $\lim_{T \rightarrow \infty} \text{Val}(s^T) = V^*$ ,  $s^T$  will be an approximately optimal policy for sufficiently large  $T$ .

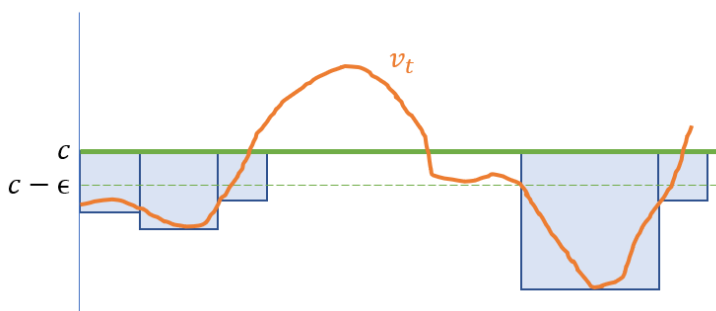
## 5 Approximate Optimality of Lazy Policies

In the previous section we solved for the optimal policy when  $z = 0$ , meaning that there is no fixed per-round cost when sampling. We now show that for general  $z$ , lazy policies are approximately optimal, obtaining at least  $1/2$  of the value of the optimal policy. Proofs appear in the full version of the paper.

We begin with a lemma that states that, for any valid sampling policy and any sequence of timesteps, it is possible to match the variance at those timesteps with a policy that only samples at precisely those timesteps, and the resulting policy will be valid.

► **Lemma 6.** *Fix any valid sampling policy  $s$  (not necessarily lazy) with resulting variances  $(v_t)$ , and any sequence of timesteps  $t_1 < t_2 < \dots < t_\ell < \dots$ . Then there is a valid policy  $s'$  such that  $\{t \mid s'_t > 0\} \subseteq \{t_1, \dots, t_\ell, \dots\}$ , resulting in a variances  $(\check{v}_t)$  with  $\check{v}_{t_i} \leq v_{t_i}$  for all  $i$ .*

The intuition is that if we take all the samples we would have spent between timesteps  $t_\ell$  and  $t_{\ell+1}$  and instead spend them all at  $t_{\ell+1}$  the result will be a (weakly) lower variance at  $t_{\ell+1}$ . We next show that any policy can be converted into a lazy policy at a loss of at most half of its value.



■ **Figure 2** Visualizing the construction in the proof of Theorem 7. Variance (vertical) is plotted against time (horizontal). We approximate the value of an optimal policy’s variance (orange) given  $c$  (green). The squares (drawn in blue) cover the gap between the curves, except possibly when  $|v_t - c| < \epsilon$  (for technical reasons). The lazy policy samples on rounds corresponding to the left edge of each square, bringing the variance to each square’s bottom-left corner.

► **Theorem 7.** *The optimal lazy policy is  $1/2$ -approximate.*

See Figure 2 for an illustration of the intuition behind the result. Consider an arbitrary policy  $s$ , with resulting variance sequence  $(v_t)$ . Imagine covering the area between  $(v_t)$  and  $c$  with squares, drawn left to right with their upper faces lying on the outside option line, each chosen just large enough so that  $v_t$  never falls below the area covered by the squares. The area of the squares is an upper bound on  $\text{Val}(s)$ . Consider a lazy policy that drops a single atom on the left endpoint of each square, bringing the variance to the square’s lower-left corner. The value of this policy covers at least half of each square. Moreover, Lemma 6 implies this policy is (approximately) valid, as it matches variances from the original policy, possibly shifted early by a constant number of rounds. This shifting can introduce non-validity; we fix this by delaying the policy’s start by a constant number of rounds without affecting the asymptotic behavior.

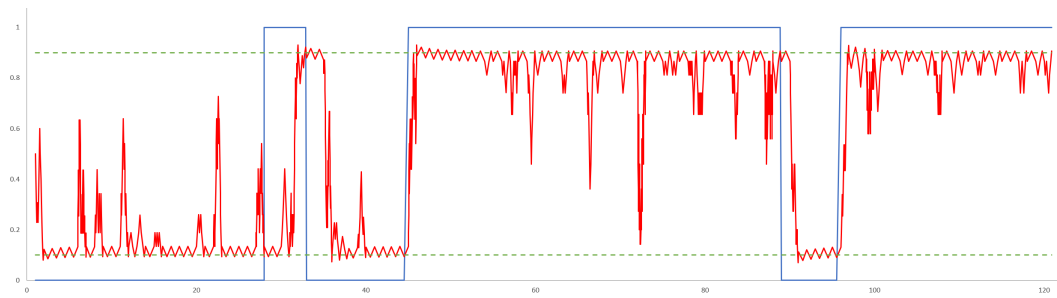
The factor  $1/2$  in Theorem 7 is tight. To see this, fix the value of  $c$  and allow the budget  $B$  to grow arbitrarily large. Then the optimal value tends to  $c$  as the budget grows, since the achievable variance on all rounds tends to 0. However, the lazy policy cannot achieve value greater than  $c/2$ , as this is what would be obtained if the variance reached 0 on the rounds on which samples are taken.

Finally, while this result is non-constructive, one can compute a policy whose value approaches an upper bound on the optimal lazy policy, in a similar manner to the optimal on-off policy. One can show the best lazy policy over any finite horizon has an “off” period (with no sampling) followed by an “on” period (where  $v_t \leq c$ ). One can then solve for the optimal number of samples to take whenever  $\tilde{v}_t > c$  by optimizing either value per unit of (fixed plus per-sample) sampling cost, or by fully exhausting the budget, whichever is better. Details appear in the full version of the paper.

## 6 Extensions and Future Directions

We describe two extensions of our model in the appendix. First, we consider a continuous-time variant where samples can be taken continuously subject to a flow cost, in addition to being requested as discrete atoms. The decision-maker selects actions continuously, and aims to minimize loss over time. All of our results carry forward to this continuous extension.

Second, returning to discrete time, we consider a non-Gaussian instance of our framework.



**Figure 3** Simulating the optimal policy for the non-Gaussian extension. The round number is on the horizontal axis. The hidden state of the world is binary and evolves stochastically (blue). The optimal policy tracks a posterior distribution over the hidden state (red), and takes samples in order to maintain a tuned level of certainty (dashed green). Note that most rounds have only a small number of samples, with occasional spikes triggered adaptively in response to uncertainty.

In this model, there is a binary hidden state of the world, which flips each round independently with some small probability  $\epsilon > 0$ . The decision-maker's action in each round is to guess the hidden state of this simple two-state Markov process, and the objective is to maximize the fraction of time that this guess is made correctly. Each sample is a binary signal correlated with the hidden state, matching the state of the world with probability  $\frac{1}{2} + \delta$  where  $\delta > 0$ . The decision-maker can adaptively request samples in each round, subject to the accumulating budget constraint, before making a guess.

In this extension, as in our Gaussian model, the optimal policy collects samples non-uniformly. In fact, the optimal policy has a simple form: it sets a threshold  $\theta > 0$  and takes samples until the entropy of the posterior distribution falls below  $\theta$ . Smaller  $\theta$  leads to higher accuracy, but also requires more samples on average, so the best policy will set  $\theta$  as low as possible subject to the budget constraint. Notably, the result of this policy is that sampling tends to occur at a slow but steady rate, keeping the entropy around  $\theta$ , except for occasional spikes of samples in response to a perceived change in the hidden state. See Figure 3 for a visualization of a numerical simulation with a budget of 6 samples (on average) per round.

More generally, whenever the state evolves in a heavy-tailed manner, it is tempting to take samples regularly in order to detect large, infrequent jumps in state value, and then adaptively take many samples when such a jump is evident. This simple model is one scenario where such behavior is optimal. More generally, can we quantify the dynamic value of data and find an (approximately) optimal data collection policy for more complex Markov chains, or other practical applications?

---

## References

- 1 Charu C. Aggarwal. A framework for diagnosing changes in evolving data streams. In *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data*, SIGMOD '03, pages 575–586, New York, NY, USA, 2003. ACM. doi:10.1145/872757.872826.
- 2 Charu C. Aggarwal. *Data Streams: Models and Algorithms (Advances in Database Systems)*. Springer-Verlag, Berlin, Heidelberg, 2006.
- 3 Charu C. Aggarwal, Jiawei Han, Jianyong Wang, and Philip S. Yu. A framework for clustering evolving data streams. In *Proceedings of the 29th International Conference on Very Large Data Bases - Volume 29*, VLDB '03, pages 81–92. VLDB Endowment, 2003. URL: <http://dl.acm.org/citation.cfm?id=1315451.1315460>.



- 4 Imanol Arrieta-Ibarra, Leonard Goff, Diego Jiménez-Hernández, Jaron Lanier, and E Glen Weyl. Should we treat data as labor? moving beyond "free". In *AEA Papers and Proceedings*, volume 108, pages 38–42, 2018.
- 5 Claudia Beleites, Ute Neugebauer, Thomas Bocklitz, Christoph Krafft, and Jürgen Popp. Sample size planning for classification models. *Analytica chimica acta*, 760:25–33, 2013.
- 6 Michael Buhrmester, Tracy Kwang, and Samuel D Gosling. Amazon's mechanical turk: A new source of inexpensive, yet high-quality, data? *Perspectives on psychological science*, 6(1):3–5, 2011.
- 7 Yiling Chen, Nicole Immorlica, Brendan Lucier, Vasilis Syrgkanis, and Juba Ziani. Optimal data acquisition for statistical estimation. In *Proceedings of the 2018 ACM Conference on Economics and Computation*, pages 27–44. ACM, 2018.
- 8 Yiling Chen and Bo Waggoner. Informational substitutes. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 239–247. IEEE, 2016.
- 9 Junghwan Cho, Kyewook Lee, Ellie Shin, Garry Choy, and Synho Do. How much data is needed to train a medical image deep learning system to achieve necessary high accuracy? *CoRR*, abs/1511.06348, 2015. [arXiv:1511.06348](https://arxiv.org/abs/1511.06348).
- 10 Corinna Cortes, Lawrence D Jackel, Sara A Solla, Vladimir Vapnik, and John S Denker. Learning curves: Asymptotic values and rate of convergence. In *Advances in Neural Information Processing Systems*, pages 327–334, 1994.
- 11 Bo Cowgill, Justin Wolfers, and Eric Zitzewitz. Using prediction markets to track information flows: Evidence from google. In *AMMA*, page 3, 2009.
- 12 Mayur Datar, Aristides Gionis, Piotr Indyk, and Rajeev Motwani. Maintaining stream statistics over sliding windows: (extended abstract). In *Proceedings of the Thirteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '02, pages 635–644, Philadelphia, PA, USA, 2002. Society for Industrial and Applied Mathematics. URL: <http://dl.acm.org/citation.cfm?id=545381.545466>.
- 13 I. Diakonikolas, G. Kamath, D. M. Kane, J. Li, A. Moitra, and A. Stewart. Robust estimators in high dimensions without the computational intractability. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 655–664, 2016.
- 14 I. Diakonikolas, D. M. Kane, and A. Stewart. Statistical query lower bounds for robust estimation of high-dimensional gaussians and gaussian mixtures. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 73–84, 2017.
- 15 Ilias Diakonikolas, Gautam Kamath, Daniel M. Kane, Jerry Li, Ankur Moitra, and Alistair Stewart. Robustly learning a gaussian: Getting optimal error, efficiently. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '18, pages 2683–2702, Philadelphia, PA, USA, 2018. Society for Industrial and Applied Mathematics. URL: <http://dl.acm.org/citation.cfm?id=3174304.3175475>.
- 16 Fang Fang, Maxwell Stinchcombe, and Andrew Whinston. "putting your money where your mouth is"-a betting platform for better prediction. *Review of Network Economics*, 6(2), 2007.
- 17 Simon Fothergill, Helena Mentis, Pushmeet Kohli, and Sebastian Nowozin. Instructing people for training gestural interactive systems. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1737–1746. ACM, 2012.
- 18 Anna C. Gilbert, Sudipto Guha, Piotr Indyk, Yannis Kotidis, S. Muthukrishnan, and Martin J. Strauss. Fast, small-space algorithms for approximate histogram maintenance. In *Proceedings of the Thiry-fourth Annual ACM Symposium on Theory of Computing*, STOC '02, pages 389–398, New York, NY, USA, 2002. ACM. doi:10.1145/509907.509966.
- 19 Shugang Hao and Lingjie Duan. Regulating competition in age of information under network externalities. *IEEE Journal on Selected Areas in Communications*, 38(4):697–710, 2020.
- 20 Panagiotis G Ipeirotis. Analyzing the amazon mechanical turk marketplace. *XRDS: Crossroads, The ACM Magazine for Students*, 17(2):16–21, 2010.
- 21 Adam Tauman Kalai, Ankur Moitra, and Gregory Valiant. Efficiently learning mixtures of two gaussians. In *Proceedings of the Forty-Second ACM Symposium on Theory of Computing*, STOC '10, page 553–562, New York, NY, USA, 2010. Association for Computing Machinery. doi:10.1145/1806689.1806765.

- 22 H. M. Kalayeh and D. A. Landgrebe. Predicting the required number of training samples. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-5(6):664–667, November 1983. doi:10.1109/TPAMI.1983.4767459.
- 23 Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45, 1960.
- 24 Sanjit Kaul, Roy Yates, and Marco Gruteser. Real-time status: How often should one update? In *2012 Proceedings IEEE INFOCOM*, pages 2731–2735. IEEE, 2012.
- 25 Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning*, pages 1885–1894, 2017.
- 26 K. A. Lai, A. B. Rao, and S. Vempala. Agnostic estimation of mean and covariance. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 665–674, Los Alamitos, CA, USA, October 2016. IEEE Computer Society. doi:10.1109/FOCS.2016.76.
- 27 Ern J Lefferts, F Landis Markley, and Malcolm D Shuster. Kalman filtering for spacecraft attitude estimation. *Journal of Guidance, Control, and Dynamics*, 5(5):417–429, 1982.
- 28 Chao Li and Gerome Miklau. Pricing aggregate queries in a data marketplace. In *WebDB*, pages 19–24, 2012.
- 29 Annie Liang, Xiaosheng Mu, and Vasilis Syrgkanis. Dynamic information acquisition from multiple sources. *arXiv preprint*, 2017. arXiv:1703.06367.
- 30 Nihar Bhadrish Shah and Denny Zhou. Double or nothing: Multiplicative incentive mechanisms for crowdsourcing. In *Advances in neural information processing systems*, pages 1–9, 2015.
- 31 Xiaohui Song and Jane W-S Liu. Performance of multiversion concurrency control algorithms in maintaining temporal consistency. In *Proceedings Fourteenth Annual International Computer Software and Applications Conference*, pages 132–133. IEEE Computer Society, 1990.
- 32 Florian Stahl, Fabian Schomm, and Gottfried Vossen. The data marketplace survey revisited. Technical report, Working Papers, ERCIS-European Research Center for Information Systems, 2014.
- 33 Amos Storkey. Machine learning markets. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 716–724, 2011.
- 34 Sebastian Thrun. Probabilistic algorithms in robotics. *Ai Magazine*, 21(4):93, 2000.
- 35 Xuehe Wang and Lingjie Duan. Dynamic pricing for controlling age of information. In *2019 IEEE International Symposium on Information Theory (ISIT)*, pages 962–966. IEEE, 2019.
- 36 Daniel B Work, Olli-Pekka Tossavainen, Sébastien Blandin, Alexandre M Bayen, Tochukwu Iwuchukwu, and Kenneth Tracton. An ensemble kalman filtering approach to highway traffic estimation using gps enabled mobile devices. In *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*, pages 5062–5068. IEEE, 2008.
- 37 Xianwen Wu, Jing Yang, and Jingxian Wu. Optimal status update for age of information minimization with an energy harvesting source. *IEEE Transactions on Green Communications and Networking*, 2(1):193–204, 2017.
- 38 Meng Zhang, Ahmed Arafa, Jianwei Huang, and H Vincent Poor. How to price fresh data. *arXiv preprint*, 2019. arXiv:1904.06899.

# Learning and Strongly Truthful Multi-Task Peer Prediction: A Variational Approach

Grant Schoenebeck 

University of Michigan, Ann Arbor, MI, USA

<http://schoeneb.people.si.umich.edu/>

schoeneb@umich.edu

Fang-Yi Yu 

Harvard University, Cambridge, MA, USA

<http://www-personal.umich.edu/~fayu/>

fangyiyu@seas.harvard.edu

---

## Abstract

Peer prediction mechanisms incentivize agents to truthfully report their signals even in the absence of verification by comparing agents' reports with those of their peers. In the detail-free multi-task setting, agents are asked to respond to multiple independent and identically distributed tasks, and the mechanism does not know the prior distribution of agents' signals. The goal is to provide an  $\epsilon$ -strongly truthful mechanism where truth-telling rewards agents "strictly" more than any other strategy profile (with  $\epsilon$  additive error) even for heterogeneous agents, and to do so while requiring as few tasks as possible.

We design a family of mechanisms with a scoring function that maps a pair of reports to a score. The mechanism is strongly truthful if the scoring function is "prior ideal". Moreover, the mechanism is  $\epsilon$ -strongly truthful as long as the scoring function used is sufficiently close to the ideal scoring function. This reduces the above mechanism design problem to a learning problem – specifically learning an ideal scoring function. Because learning the prior distribution is sufficient (but not necessary) to learn the scoring function, we can apply standard learning theory techniques that leverage side information about the prior (e.g., that it is close to some parametric model). Furthermore, we derive a variational representation of an ideal scoring function and reduce the learning problem into an empirical risk minimization.

We leverage this reduction to obtain very general results for peer prediction in the multi-task setting. Specifically,

**Sample Complexity.** We show how to derive good bounds on the number of tasks required for different types of priors—in some cases exponentially improving previous results. In particular, we can upper bound the required number of tasks for parametric models with bounded learning complexity. Furthermore, our reduction applies to myriad continuous signal space settings. To the best of our knowledge, this is the first peer-prediction mechanism on continuous signals designed for the multi-task setting.

**Connection to Machine Learning.** We show how to turn a soft-predictor of an agent's signals (given the other agents' signals) into a mechanism. This allows the practical use of machine learning algorithms that give good results even when many agents provide noisy information.

**Stronger Properties.** In the finite setting, we obtain  $\epsilon$ -strongly truthful mechanisms for any stochastically relevant prior. Prior works either only apply to more restrictive settings, or achieve a weaker notion of truthfulness (informed truthfulness).

**2012 ACM Subject Classification** Information systems → Incentive schemes; Theory of computation → Quality of equilibria; Mathematics of computing → Variational methods; Mathematics of computing → Information theory

**Keywords and phrases** Information elicitation without verification, crowdsourcing, machine learning

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.78

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/2009.14730>.

**Funding** *Grant Schoenebeck*: National Science Foundation 1618187 and 2007256.

*Fang-Yi Yu*: National Science Foundation 1618187 and 2007256.



© Grant Schoenebeck and Fang-Yi Yu;

licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 78; pp. 78:1–78:20



Leibniz International Proceedings in Informatics

LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

## 1 Introduction

Peer prediction is the problem of information elicitation without verification. Peer prediction mechanisms exploit the interdependence in agents' signals to incentive agents to report their private signal truthfully even when the reports cannot be directly verified. In *the multi-task setting* [5], each agent is asked to respond to multiple, independent tasks. For example:

► **Example 1 (Commute time).** We can collect data from drivers to estimate the commute time of a certain route. Each driver's daily commute time might be modeled in the following way: each day, the route has an expected time generated from a Gaussian distribution, and each driver's commute time is the expected time perturbed by independently distributed Gaussian noise.

Peer prediction from strategic agents has been attracting a surge of interest in economics and computer science. Several previous works [1, 16, 19] can be understood as using particular learning algorithms to learn nice payment functions that capture the interdependence in agents' reports. In this paper, we decouple these two components: mechanism design and learning algorithms. This framework provides a clean black-box reduction from learning algorithms to peer prediction mechanism.

One advantage of our framework is that we can use results from machine learning about complexity of learning parameters of priors to obtain bounds on the sample complexity (number of tasks required) of our mechanism. For instance, using our reduction, we can easily exponentially improve the required number of tasks in the previous work [28].

Two features of our mechanisms enable us to work in more complicated settings. First, our mechanisms use mutual information to pay agents. This allows us to use aggregation algorithms and pay an agent the mutual information between her reports and the aggregated outcome of the other agents. For example, suppose the agents' report's average quality is low, and a large fraction of agents report random noise. In that case, we can use aggregation to enhance the signal to noise ratio and provide a robust incentive to strategic workers. The second feature of our mechanisms is a variational formulation, which ensures one-sided error such that we can only underestimate the mutual information but not overestimate it. Thus, we can use deep learners or other rich enough functions to learn a good payment in practice.

In addition to the above contributions, we also improve previous work in two axes: the *truthfulness guarantee* and the *prior assumption*.

The truthful guarantee explains *how good the truth-telling strategy is in the mechanism* (formally defined in Sect.2.1). Is truth-telling always the best response regardless of other's strategy (dominantly truthful)? Or is truth-telling a Bayesian Nash equilibrium in which agents receive strictly higher payment than any other non-permutation equilibrium (strongly truthful) where a permutation equilibrium is one where agents report a permutation of the signals? A slightly weaker property is *informed truthful* where no strategy profile pays strictly more than truth-telling, and truth-telling pays more than any uninformative equilibrium. Our pairing mechanisms is dominantly truthful if the number of tasks is infinite and approximately strongly truthful when the number of tasks is finite.

Another axis upon which to measure a peer prediction mechanism's performance is its assumption on the prior of agents' signals. There are two motivations to understand how general the prior can be. First, in practice, we need a peer prediction mechanism that works for general settings, e.g., continuous signals in the aforementioned commute time example. Second, a mechanism's prior assumption often reveals why the mechanism works. Thus, improving prior assumptions can push our theoretical understanding of peer prediction mechanisms.

It is well-known that a necessary condition for the truth-telling strategy profile to be a strict Bayesian Nash equilibrium is that agents' signals need to be *stochastic relevant* (Definition 5) [32]. However, when is stochastic relevance a sufficient condition? Previous multi-task peer prediction mechanisms make ad hoc assumptions on agents' private signals (positively correlated [5], fine-grained [16], strictly correlated [11], or latent variable models [19]) which are discussed in Sect. 2.2. This restricts the settings in which they can be used. Moreover, all the above mechanisms only work when agents' signals are in a finite space<sup>1</sup>.

In this paper, we show stochastic relevance is also a sufficient condition in the multi-tasks setting. Our pairing mechanisms are approximately-strongly truthful as long as the prior is stochastic relevant. In particular, the space of agents' signals can be countably infinite or even continuous. To the authors' knowledge, our mechanism is the first (detail-free) multi-task mechanism that works all stochastically relevant priors.

Besides the above properties, we also require our mechanisms 1) are *minimal* which only elicit the agents' signals and no additional information; 2) are *detail-free* which do not require foreknowledge of the prior; and 3) have *low sample number*, where each agent only needs to answer a few questions for the mechanism to achieve approximately strong truthfulness.

**Our Techniques.** Prior work [16] has shown that paying agents according to the  $\Phi$  mutual information (a generalization of the Shannon mutual information) between their signals is a good idea. This is because, if agents try to strategically manipulate their signals, the  $\Phi$  mutual information can only decrease. However, a key open question is how to compute the mutual information while having access to only a few signals for each agent. Moreover, the computation needs to be done in a way that maintains the incentive guarantees of the mechanism.

We solve this issue. First, we convert the mechanism design problem into an optimization problem (Theorem 16). The  $\Phi$  mutual information of a pair of random variables can be defined as the  $\Phi$  divergence between two distributions: the joint distribution and the product of marginal distributions. The  $\Phi$  divergence is just a measure of distance between the two distributions and contains the KL-divergence as a special case. The problem of computing the  $\Phi$  divergence, using variational representation as a bridge, can be changed into the optimization problem of finding the best “distinguisher” between these two distributions. We call such a distinguisher a *scoring function*. The optimal scoring function (distinguisher) can differentiate the two distributions with a score equal to the  $\Phi$  divergence, whereas any other scoring function (distinguisher) yields a lower score. Thus, once one has this optimal scoring function, estimating the  $\Phi$  divergence (and hence  $\Phi$  mutual information) is easy – just compute its score. In this paper we call the optimal scoring function for a particular prior  $P$ , the  $(P, \Phi)$ -ideal scoring function which can be easily computed when the prior  $P$  is known.

Our mechanism will reward agents according to some scoring function. Importantly, agents' ex-ante payments under prior  $P$  are maximized when *both* the distinguisher used is the  $(P, \Phi)$ -ideal scoring function, and the agents are truth-telling. Consequently, if we already have the  $(P, \Phi)$ -ideal scoring function, the mechanism incentivizes truthful reporting. Additionally, agents will receive a smaller payout if the mechanism fails to find the optimal scoring function. Thus agents are naturally incentivized to aid the mechanism in finding it and cannot gain by deceiving the mechanism into using a suboptimal scoring function.

---

<sup>1</sup> Discretization approach is not practical in most situations. See [17] for more discussion.

Compared with Kong and Schoenebeck [16], our variational characterization provides a better truthfulness guarantee when the number of tasks is finite. We can uniformly upper bound the ex-ante payments under any non-truthful strategy profile (Definition 4) even when the learning algorithm cannot estimate the ideal scoring functions under those non-truthful strategies. This property is vital for continuous signal spaces where agents may adversarially adopt the worst possible strategy profiles to compromise the learning algorithm.

The above observations transform the problem from designing a mechanism to simply learning the  $(P, \Phi)$ -ideal scoring function given samples from a prior. We provide two algorithms to learn the scoring function. The first one is a *generative* approach which estimates the whole density function of the prior and computes a scoring function from it. In a *discriminative* approach, we formulate the estimation of the ideal scoring function as a convex optimization problem, empirical risk minimization [22], and estimate the scoring function directly. This latter approach allows us to use state-of-art convex optimization solvers to estimate good scoring functions.

**Our Contributions.** In this paper, we leverage the above insights to design a  $\Phi$ -pairing mechanism that is minimal and detail-free for heterogeneous agents. In particular:

**Sample Complexity.** We show how to derive good bounds on the number of tasks required for different types of priors—in some cases exponentially improving previous results. In particular, we can upper bound the required number of tasks for parametric models with bounded learning complexity (as measured by a continuous analog of the VC dimension). Furthermore, our reduction applies to myriad continuous signal space settings. To the best of our knowledge, this is the first peer-prediction mechanism on continuous signals designed for the multi-question setting.

**Connections to Machine Learning.** In this paper, we discuss how to convert information elicitation design into three learning problems. 1) The first one is a *generative* approach which estimates the whole density function of the prior and computes a scoring function from it. 2) We formulate the estimation of the ideal scoring function as a convex optimization problem, empirical risk minimization, and estimate the scoring function directly. 3) Finally, we show how to turn a soft-predictor of an agent’s signals (given the other agents’ signals) into a mechanism. This allows the practical use of machine learning algorithms that give good results even when many agents provide noisy information.

**Stronger Properties.** In the finite setting, we obtain  $\epsilon$ -strongly truthful mechanisms for any stochastically relevant prior. Prior works either only apply to more restrictive settings [11], or achieve a weaker notion of truthfulness (informed truthfulness) [28, 1].

## 1.1 Related Work

**Multi-task setting.** In the multi-task setting, Dasgupta and Ghosh [5] propose a *strongly truthful* mechanism when the signal space is binary and every pair of agents’ signals are assumed to be positively correlated. Both Kong and Schoenebeck [16] and Shnayder et al. [28] independently generalize Dasgupta and Ghosh [5] to discrete signal spaces, though in different manners illustrated as follows.

Kong and Schoenebeck [16] present the  $\Phi$ -**mutual information mechanism**, a multi-task peer prediction mechanism for the finite signal space setting with arbitrary interdependence between signals. Unfortunately, the sample number is infinite. They show that their mechanism is strongly truthful as long as the prior is “fine-grained” where, roughly speaking, no two signals can be interpreted as different names for the same signal. To define their mechanism they introduce the notion of  $\Phi$ -mutual information (of which Shannon mutual



■ **Table 1** Comparison to previous work.

	D&G [5]	CA [28, 1]	$\Phi$ -MIM [16]	DMI [11]	$\Phi$ -pairing mechanism
Signal space	binary	finite	finite	finite	continuous
Prior Assumptions	positive correlated	stochastic relevant	fine -grained	strictly correlated	stochastic relevant
Truthful	✓	✓	✓	✓	✓
Informed-truthful	✓	✓	✓	✓	✓
Strongly truthful	✓		✓	✓	✓
Detail-free	✓	✓	✓	✓	✓

information is a special case) where  $\Phi$  is any convex function. Their mechanism pays each agent the  $\Phi$ -mutual information between her reports and the reports of another randomly chosen agent. Strategic behavior is shown to not increase  $\Phi$ -mutual information by a generalized version of the data processing inequality. Unfortunately, their analysis requires infinite sample number to measure this  $\Phi$ -mutual information and does not handle errors in estimation.

Shnayder et al. [28] introduce the **Correlated Agreement (CA) mechanism** which also generalizes Dasgupta and Ghosh [5] to any finite signal space. On the one hand, the CA mechanism can assume the knowledge of the “signal structure” (which tells which signals are positively and negatively correlated). In this case they can provide a mechanism that is truthful with sample number of two.<sup>2</sup> On the other hand, when agents are homogeneous the CA mechanism can learn the signal structure, albeit with some chance of error. The CA mechanism is shown to be robust to this error, and is  $\epsilon$ -informed truthful (a slightly weaker notion than strongly truthful). Agarwal et al. [1] extend the above work [28] to a particular setting of heterogeneous agents where agents are (close to) one of a fixed number of types. They establish a  $O(n)$  sample number in this new setting where  $n$  is the number of agents.

Note that in the above works, a new robustness (error) analysis is required for each different setting of interdependence between signals. Interestingly, the CA mechanism can be viewed as a special case of the aforementioned  $\Phi$ -mutual information mechanism using the total variation distance mutual information (i.e.,  $\Phi(a) = |a - 1|/2$ ). However, instead of directly computing this mutual information, the CA mechanism obtains a consistent estimator of it [16]. Similarly, in the special case that our mechanism implements the total variation distance, we also recover the CA mechanism. However, our analysis is entirely different.

Kong [11] shows an elegant way of obtaining strongly truthful mechanisms (DMI mechanism) for the multitask setting. Our results are incommensurate with these results. In our results, the sample complexity grows with the  $\epsilon$  in the desired  $\epsilon$ -strongly truthful guarantee but is independent of the number of signals. In Kong [11], there is an exact strongly truthful guarantee but sample complexity grows in the size of the signal space. However, the prior structure needs to be strictly correlated, which is a stronger assumption than stochastic relevance. We provide a comparison in Table 1 and Sect. 2.2. In particular, her mechanism requires all agents’ report spaces are finite and have the same size. This restricts the application of an aggregation algorithm (as mentioned in the introduction and Sect. 7).

<sup>2</sup> The original paper shows it requires 3, but it actually only needs 2 tasks.



**Single task setting.** In general, agents do not (necessarily) have multiple identical and independent signals. Without this property, most of the mechanisms require knowledge of a common prior (not detail-free) or for agents to report their whole posterior distribution of other’s signals (not minimal). The later solution is especially difficult to apply to complicated signal spaces (e.g. asking agents to report their probability density function of others’ continuous signals).

Miller et al. [20] introduce the peer prediction mechanism which is the first mechanism that has truth-telling as a strict Bayesian Nash equilibrium and does not need verification. However, their mechanism requires the full knowledge of the common prior and there exist some equilibria that are paid more than truth-telling. In particular, the oblivious equilibrium pays strictly more than truth-telling. Kong et al. [12] modify the original peer prediction mechanism such that truth-telling pays strictly better than any other equilibrium but still requires the full knowledge of the common prior. Prelec [23] designs the first detail-free peer prediction mechanism – Bayesian truth serum (BTS) in the one question setting. Several other works study the one-question setting of BTS [24, 25, 31, 14, 27]. For continuous signals, Radanovic and Faltings [25] apply a discretization approach and use a new payment method, but that is also non-minimal. Goel and Faltings [10] work on a mixture of normal distributions with an infinite number of agents.

**Miscellany.** Liu and Chen [18] design a peer prediction mechanism where each agents’ responses are not compared to another agents’, but rather the output of a machine learning classifier that learns from all the other agents’ responses. Liu and Chen [19] design a non-minimal approximate dominant strategy mechanism that uses surrogate loss functions as tools to correct for the mistakes in agents’ reports. Kong and Schoenebeck [15] studies the related goal for forecast elicitation, and like the present work uses Fenchel’s duality to reward truth-telling (though in a different manner).

One interesting, but orthogonal, line of work looks at “cheap” signals, where agents can coordinate on less useful information. For example, instead of grading an assignment based on correctness, a grader could only spot check the grammar. Gao et al. [9] introduces the issue, while Kong and Schoenebeck [13] shows a partial solution using conditional mutual information.

The recent book [7] surveys additional results from this area.

## 1.2 Structure of Paper

Sect. 2 introduces some basic notions. In particular, Sect. 2.2 defines scoring functions, which will play an important role in this paper.

At the beginning of Sect. 3, we define a central component of our  $\Phi$ -pairing mechanism, Mechanism 1, which takes agents’ report and a scoring function  $K$  as input. In Sect. 4, we consider the full information setting. We show, in the Mechanism 1 with an ideal scoring function, agents are incentivized to report their signals truthfully. In Sect. 5, we prove Theorem 11, and main technical lemmas. In Sect. 6, we define a notion of approximation of an ideal scoring function and introduce our framework that reduces the mechanism problem for information elicitation to a learning problem for an ideal scoring function (Theorem 16). In Sect. 6.3, we focus on the learning problem introduced in Sect. 6. We first show two sufficient conditions for approximating an ideal scoring function in Sect. 6.3.1. Then, we present two algorithms to derive approximately ideal scoring functions from agents’ reports in Sect. 6.3.2.

In Sect. 7, we generalize Mechanism 1 to more than two agents. We show how machine learning techniques can be naturally integrated with our mechanism.

## 2 Preliminaries

We use  $(\Omega, \mathcal{F}, \mu)$  to denote a measure space where  $\mathcal{F}$  is a  $\sigma$ -algebra on the outcome space  $\Omega$  and  $\mu$  is a measure. Let  $\Delta_\Omega$  denote the set of distributions of over  $(\Omega, \mathcal{F})$ ,<sup>3</sup> and  $\mathcal{P}$  as a subset of distributions in  $\Delta_\Omega$ . Given a distribution  $P$ , we also use  $P$  to denote the density function where  $P(\omega)$  is the probability density of outcome  $\omega \in \Omega$ . We use uppercase for a random object  $X$  and lowercase for the outcome  $x$ . In this paper we consider  $\Phi$  to be a convex continuous function and use  $\text{dom}(\Phi)$  to denote its domain.

### 2.1 Mechanism Design for Information Elicitation

For simplicity we first consider two agents, Alice and Bob, who work on a set of  $m$  tasks denoted as  $[m]$ . For each task  $s \in [m]$ , Alice receives a signal  $x_s$  in  $\mathcal{X}$  and Bob a signal  $y_s$  in  $\mathcal{Y}$ . We use  $(\mathbf{X}, \mathbf{Y}) \in (\mathcal{X} \times \mathcal{Y})^m$  to denote the *signal profile* of Alice and Bob which is generated from a prior distribution  $\mathbb{P}$ . In this paper, we make the following assumption:

► **Assumption 2** (A priori similar tasks [5]).  *$\mathbb{P}$  is a prior, and each task is identically and independently (i.i.d.) generated: there exists a distribution  $P_{X,Y}$  over  $\mathcal{X} \times \mathcal{Y}$  such that  $\mathbb{P} = P_{X,Y}^m$ . Moreover, we assume the marginal distributions have full supports,  $P_X(x) > 0$  and  $P_Y(y) > 0$  for all  $x \in \mathcal{X}$  and  $y \in \mathcal{Y}$ .*

Given a report profile of Alice,  $\hat{\mathbf{x}} \in \mathcal{X}^m$  and Bob,  $\hat{\mathbf{y}} \in \mathcal{Y}^m$ , an *information elicitation mechanism*  $\mathcal{M} = (M_A, M_B)$  with  $m$  tasks pays  $M_A(\hat{\mathbf{x}}, \hat{\mathbf{y}}) \in \mathbb{R}$  to Alice, and  $M_B(\hat{\mathbf{x}}, \hat{\mathbf{y}}) \in \mathbb{R}$  to Bob. In the rest of the paper we often only define notions for Alice, and define Bob's in the symmetric way.

Besides Assumption 2, we assume their strategies are uniform and independent across different tasks which is also made in previous work [5, 28, 16].

► **Assumption 3** (Uniform strategy). *Formally, the strategy of Alice is a random function  $\theta_A : \mathcal{X} \rightarrow \Delta_{\mathcal{X}}$  where  $\theta_A(x, \hat{x})$  is the probability that Alice reports  $\hat{x}$  conditioning on her private information  $x$ . That is, each report only depends on the corresponding signal.*

For instance, given Alice receiving  $\mathbf{x} \in \mathcal{X}^m$  the probability that Alice reports  $\hat{\mathbf{x}} \in \mathcal{X}^m$  is  $\Pr[\hat{\mathbf{X}} = \hat{\mathbf{x}}] = \prod_{s \in [m]} \theta_A(x_s, \hat{x}_s)$ . We call  $\boldsymbol{\theta} = (\theta_A, \theta_B)$  a the *strategy profile*. The *ex-ante payment* to Alice under a strategy profile  $\boldsymbol{\theta}$  and a prior  $\mathbb{P}$  in mechanism  $\mathcal{M}$  is  $u_A(\boldsymbol{\theta}; \mathbb{P}, \mathcal{M}) \triangleq \mathbb{E}_{(\mathbf{X}, \mathbf{Y})} [\mathbb{E}_{(\hat{\mathbf{X}}, \hat{\mathbf{Y}})} [M_A(\hat{\mathbf{x}}, \hat{\mathbf{y}})] \mid (\mathbf{x}, \mathbf{y})]$  where we use a semicolon to separate the variable,  $\boldsymbol{\theta}$ , and parameters  $\mathbb{P}$  and  $\mathcal{M}$ . Note that a strategy profile  $\boldsymbol{\theta}$  can be seen as a Markov operator on the signal space  $\mathcal{X} \times \mathcal{Y}$ , so that Alice and Bob's reports,  $\boldsymbol{\theta} \circ P$ , is also a distribution on the signal space  $\mathcal{X} \times \mathcal{Y}$ .

In the literature on peer-prediction, there are three important classes of strategies. We use  $\boldsymbol{\tau}$  to denote the **truth-telling strategy profile** where both agents' reports are equal to their private signals with probability 1, e.g., Alice's strategy is  $\tau_A(x, \hat{x}) = \mathbb{I}[x = \hat{x}]$ . A strategy profile is a **permutation strategy profile** if both agents' strategy are a (deterministic) permutation, a bijection between signals and reports. Finally, a strategy profile is **oblivious** or *uninformed* if even one of the agents' strategies does not depend on their signal: that is for Alice  $\theta_A(x, \hat{x}) = \theta_A(x', \hat{x})$  for all  $x, x'$ , and  $\hat{x}$  in  $\mathcal{X}$ . Note that the set of permutation strategy profiles includes the truth-telling strategy profile  $\boldsymbol{\tau}$  but does not include any oblivious strategy profiles.

<sup>3</sup> We assume these distribution has a density function with respect to the  $\mu$ ,  $P \ll \mu$  for all  $P \in \Delta_\Omega$ . The distributions in  $\Delta_\Omega$  depend on  $\mathcal{F}$  and  $\mu$ , but we omit it to simplify the notation. The density is defined as the Radon–Nikodym derivative  $\frac{dP}{d\mu}$  which exists because  $P$  is dominated by  $\mu$ .

**Truthful Guarantees.** We now define some truthfulness guarantees for our mechanism  $\mathcal{M}$  that differ in how unique the high payoff of truth-telling strategy profile is:

**Truthful:** the truth-telling strategy profile  $\tau$  is a Bayesian Nash Equilibrium, and has the highest payment to both Alice and Bob.

**Informed-truthful [28]:** Truthful and also for each agent  $\tau$  is strictly better than any oblivious strategy profiles. For any oblivious strategy profile  $\theta$ ,  $u_A(\tau; \mathbb{P}, \mathcal{M}) > u_A(\theta; \mathbb{P}, \mathcal{M})$  and  $u_B(\tau; \mathbb{P}, \mathcal{M}) > u_B(\theta; \mathbb{P}, \mathcal{M})$ .

**Strongly truthful [28, 16]:** Truthful and also for each agent  $\tau$  is strictly better than all non-permutation strategy profiles. For any non-permutation strategy profile  $\theta$ ,  $u_A(\tau; \mathbb{P}, \mathcal{M}) > u_A(\theta; \mathbb{P}, \mathcal{M})$  and  $u_B(\tau; \mathbb{P}, \mathcal{M}) > u_B(\theta; \mathbb{P}, \mathcal{M})$ .

**Dominant truthful:** Each agent report truthfully leads to higher expected payoff than other strategies, regardless of other agent's reporting strategies. For any strategy profile  $\theta$ , we have  $u_A(\tau; \mathbb{P}, \mathcal{M}) > u_A(\theta; \mathbb{P}, \mathcal{M})$  and  $u_B(\tau; \mathbb{P}, \mathcal{M}) > u_B(\theta; \mathbb{P}, \mathcal{M})$ .

We can also call a general mapping truthful, informed-truthful, strongly truthful, dominant truthful when it satisfy the corresponding property.

In this work, we consider an approximate version of above statements with low sample number. For example, given  $\epsilon > 0$ , a mechanism  $\mathcal{M}$  with  $m(\epsilon)$  tasks (the sample number)<sup>4</sup> is  $\epsilon$ -strongly truthful with  $m(\epsilon)$  tasks if there exists a mapping from strategy profiles to ex-ante payments such that 1) this mapping is strongly truthful; 2) for all  $\epsilon$  the ex-ante payments of our mechanism with  $m(\epsilon)$  tasks is within  $\epsilon$  of this mapping.

Now we define the sample number for approximately truthfulness guarantees.

► **Definition 4.** Given a family of joint signal distributions  $\mathcal{P}$  and a function  $S : \mathbb{R}_{>0} \rightarrow \mathbb{N}$  we say a mechanism  $\mathcal{M}$  is  $\epsilon$ -strongly truthful on  $\mathcal{P}$  with  $S(\epsilon)$  number of tasks, if there exists a strongly truthful mapping  $F = (F_A, F_B)$  from joint signal distributions and strategy profiles to payments such that for all  $\epsilon > 0$  and  $m \geq S(\epsilon)$

- the ex-ante payment under the truth-telling strategy profile in  $\mathcal{M}$  with  $m$  number of tasks is within  $\epsilon$  additive error from  $F$ : for all  $P \in \mathcal{P}$ ,  $u_A(\tau; P, \mathcal{M}) \geq F_A(\tau, P) - \epsilon$ ;
- and the ex-ante payment under any strategy profile  $\theta$  in  $\mathcal{M}$  with  $m$  number of tasks is bounded above by  $F$ : for all  $P \in \mathcal{P}$ , and  $\theta$ ,  $u_A(\theta; P, \mathcal{M}) \leq F_A(\theta, P)$ .

And the inequality also holds for Bob's ex-ante payment. Furthermore, we say  $\mathcal{M}$  is  $(\delta, \epsilon)$ -strongly truthful on  $\mathcal{P}$  with  $S(\delta, \epsilon)$  if the above conditions holds with probability  $1 - \delta$  for all  $\delta \in (0, 1)$  and  $\epsilon > 0$ . Additionally, we say  $\mathcal{M}$  is  $\epsilon$ -informed-truthful ( $\epsilon$ -truthful) with  $S(\epsilon)$  number of tasks if it is  $\epsilon$  close to an informed-truthful (truthful) mapping.

Note that our notion of  $\epsilon$ -truthfulness guarantee is quite strong. In particular, the second item requires for any strategy profile  $\theta$ , the ex-ante payment is upper bounded by a strongly truthful (informed-truthful, truthful) mapping.

## 2.2 Prior Assumptions

There are two axes to compare these peer prediction mechanism: *truthful guarantee* and *prior assumption*. Truthful guarantee asks how good the truth-telling strategy is. Prior assumption addresses how general these mechanisms are. We first introduce the weakest possible notion of interdependence that we used in our paper. Then we survey other notions proposed in previous works. Finally, we provide concrete examples to show the distinction between those notions of interdependence.

<sup>4</sup> Here mechanism which can take different length of report  $m$ . Or we can consider a family of mechanisms  $(\mathcal{M}_m)$  parameterized by the sample number (the number of tasks)  $m$ .

► **Definition 5** (Stochastic Relevant [28]). *We call  $P_{X,Y}$  stochastic relevant if for any two distinct signals  $x, x' \in \mathcal{X}$   $P_{X,Y}[Y | X = x] \neq P_{X,Y}[Y | X = x']$ . That is, Alice's posteriors on Bob's signals are different when Alice receives signal  $x$  or  $x'$ . And symmetrically, the same holds for Bob's posterior on Alice's signals.*

Stochastic relevancy is the weakest assumption we can hope for designing peer prediction mechanisms. Proposition 6 shows that if agent's signal are not stochastic relevant an agent can always misreport regardless other agents' reports even if the mechanism knows the information structure.

► **Proposition 6** (Elicitability [32]). *If the prior  $P_{X,Y}$  is not stochastic relevant, there is no mechanism that has truth-telling as a strict Bayesian Nash equilibrium.*

Besides the above notion, previous peer prediction mechanisms make ad hoc assumptions on agents' private signals.

Kong and Schoenebeck [16] studies *fine-grained* joint distributions. A joint distribution  $P_{X,Y}$  is *fine-grained* if for any distinct pairs of signals  $(x, y)$  and  $(x', y')$ ,  $\frac{P_{X,Y}(x, y)}{P_X(x)P_Y(y)} \neq \frac{P_{X,Y}(x', y')}{P_X(x')P_Y(y')}$ . Kong [11] considers *strictly correlated* distributions. A joint distribution  $P$  on a finite space  $\mathcal{X}^2$  is strictly correlated if the determinant of distribution  $P \in \mathbb{R}^{|\mathcal{X}| \times |\mathcal{X}|}$  is nonzero. Those two notions are both stronger than stochastic relevance.

### 2.3 Convex Analysis and $\Phi$ -divergence

Informally,  $\Phi$ -divergences quantify the difference between a pair of distributions over a common measurable space.

► **Definition 7** ( $\Phi$ -divergence [3, 21, 2]). *Let  $\Phi : [0, \infty) \rightarrow \mathbb{R}$  be a convex function with  $\Phi(1) = 0$ . Let  $P$  and  $Q$  be two probability distributions on a common measurable space  $(\Omega, \mathcal{F})$ . The  $\Phi$ -divergence of  $Q$  from  $P$  where  $P \ll Q$  is defined as  $D_\Phi(P||Q) \triangleq \mathbb{E}_Q[\Phi(P/Q)]$ .<sup>5</sup>*

We can use these divergences to measure interdependency between two random variables  $X$  and  $Y$ . Formally, Let  $P_{X,Y}$  be a distribution over  $(x, y) \in \mathcal{X} \times \mathcal{Y}$ , and  $P_X$  and  $P_Y$  be marginal distributions of  $X$  and  $Y$  respectively. We set  $P_X P_Y$  be the tensor product between  $P_X$  and  $P_Y$  such that  $P_X P_Y(x, y) = P_X(x)P_Y(y)$ . We call  $D_\Phi(P_{X,Y}||P_X P_Y)$  the  $\Phi$ -mutual information between  $X$  and  $Y$ .

Given a joint distribution  $P_{X,Y}$ , let **joint to marginal product ratio** at  $(x, y)$  on  $P_{X,Y}$  be  $JP_P(x, y) := \frac{P_{X,Y}(x, y)}{P_X(x)P_Y(y)}$  which is ratio between joint probability divided by the product of the probabilities at  $(x, y)$ . We will omit subscript  $P$  when there is no ambiguity. This ratio widely studied. For instance, it's called *observed to expected ratio* in life sciences literature, or *lift* in data mining for binary random variable. Additionally,  $\log JP(x, y)$  is called point-wise mutual information. Finally, note that  $\Phi$  mutual information is the average of joint to marginal product ratio applied to  $\Phi$ .

Now, we introduce some basic notions in convex analysis [26]. Let  $\Phi : [0, +\infty) \rightarrow \mathbb{R}$  be a convex function. The *convex conjugate*  $\Phi^*$  of  $\Phi$  is defined as:  $\Phi^*(b) = \sup_{a \in \text{dom}(\Phi)} \{ab - \Phi(a)\}$ . Moreover  $\Phi = \Phi^{**}$  if  $\Phi$  is continuous.

By Young-Fenchel inequality [8], we can rewrite the  $\Phi$ -divergence of  $Q$  from  $P$  in a variational form. This formulation is important to understand our mechanisms.

<sup>5</sup>  $P/Q$  is the Radon-Nikodym derivative between measures  $P$  and  $Q$ , and it is equal to the ratio of density function.

► **Theorem 8** (Variational representation [22]).

$$D_\Phi(P\|Q) = \sup_{k:\Omega \rightarrow \text{dom}(\Phi^*)} \left\{ \mathbb{E}_{\omega \sim P}[k(\omega)] - \mathbb{E}_{\omega \sim Q}[\Phi^*(k(\omega))] \right\},^6$$

and the equality holds  $D_\Phi(P\|Q) = \mathbb{E}_{\omega \sim P}[k(\omega)] - \mathbb{E}_{\omega \sim Q}[\Phi^*(k(\omega))]$  if and only if  $k \in \partial\Phi(P/Q)$  almost everywhere on  $Q$ .<sup>7</sup>

## 2.4 Scoring Function

Our constructions and analysis will make heavy use of the following functionals – scoring functions.

► **Definition 9** (Scoring function). A **scoring function**  $K : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$  is a functional (real-valued function) that maps from a pair of reports to a real value. Given a convex function  $\Phi$ , a scoring function  $K_{P,\Phi}^*$  is a  $(P_{X,Y}, \Phi)$ -**ideal scoring function** if

$$K_{P,\Phi}^*(x, y) \in \partial\Phi\left(\frac{P_{X,Y}(x, y)}{P_X(x)P_Y(y)}\right) = \partial\Phi(\text{JP}_P(x, y)). \quad (1)$$

We will use  $P$  and  $P_{X,Y}$  interchangeably later, and say  $K^*$  is ideal without specifying  $P$  and  $\Phi$  when it's clear.

A  $(P, \Phi)$ -ideal scoring function is the joint to marginal product ratio applied to  $\partial\Phi$  which is a monotone increasing function if  $\Phi$  is differentiable. joint to marginal product ratio encodes the signal structure of  $P_{X,Y}$  which measure how interdependent  $x$  and  $y$  is. Alternatively, the scoring function serves as a “distinguisher” which tries to decide whether a pair of reports came from the joint distribution or the product of the marginal distributions.

Furthermore, the ideal scoring function can also be easily computed from the density function  $P_{X,Y}$ .

## 2.5 Functional Complexity

In this section, we provide some standard notions to characterize the complexity of learning functionals which are standard [29, 30]. We will use these notions to characterize the complexity of learning an ideal scoring function.

Let  $\mathcal{K}$  be a pre-specified class of functionals  $k : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$ . Given  $k \in \mathcal{K}$ ,  $L > 0$ , and a distribution  $P_{X,Y}$ , we define the *Bernstein norm* as  $\rho_L^2(k; P) \triangleq 2L^2 \mathbb{E}_P[\exp(|k|/L) - 1 - |k|/L]$ , and  $\rho_L(\mathcal{K}; P) \triangleq \sup_{k \in \mathcal{K}} \rho_L(k, P)$ . Let  $\mathcal{N}_{[],L}(\delta, \mathcal{K}, P)$  be the smallest value of  $n$  for which there exists  $n$  pairs of functions  $\{(k_j^L, k_j^U)\}$  such that 1)  $\rho_L(k_j^U - k_j^L; P) \leq \delta$  for all  $j$  and 2) for all  $k \in \mathcal{K}$  there is a  $j$ ,  $k_j^L(x, y) \leq k(x, y) \leq k_j^U(x, y)$  for all  $(x, y) \in \mathcal{X} \times \mathcal{Y}$ . Then  $\mathcal{H}_{[],L}(\delta, \mathcal{K}, P) \triangleq \log \mathcal{N}_{[],L}(\delta, \mathcal{K}, P)$  is called the *generalized entropy with bracketing*. We further define the entropy integral as  $J_{[],L}(R, \mathcal{K}, P) \triangleq \int_0^R \sqrt{\mathcal{H}_{[],L}(u, \mathcal{K}, P)} du$ .

Our results will show that constant number of questions suffice as long as the ideal scoring functions is in some bounded complexity space  $\mathcal{K}$  where  $J_{[],L}(R, \mathcal{K}, P)$  and  $\rho_L(\mathcal{K}; P)$  are bounded.

<sup>6</sup> The sup is taken over  $k$  with finite  $\mathbb{E}_{\omega \sim P}[k(\omega)]$  and  $\mathbb{E}_{\omega \sim Q}[\Phi^*(k(\omega))]$ .

<sup>7</sup>  $\partial\Phi$  is the subgradient of  $\Phi$ , and the formal definition can be found in [26]. Here we only use the equality condition when  $\Omega$  is finite.

### 3 $\Phi$ -Divergence Pairing Mechanisms

In this section, we first define a class of multi-task peer-prediction mechanisms  $\mathcal{M}^{\Phi, K}$ . The mechanism is parametrized by a convex function  $\Phi$  and a scoring function  $K$  (Definition 9). Then we briefly discuss how to obtain a good scoring function, and develop algorithms for estimating good scoring function.

The process of this mechanism is quite simple. Given a scoring function  $K$  and  $\Phi$ , we arbitrarily choose one task  $b$ , and two distinct tasks  $p$  and  $q$  from  $m \geq 2$  tasks. Alice gets paid by Eqn. (2) the scoring function on her and Bob's reports on task  $b$  minus the  $\Phi^*$  applied to the scoring function on her report on  $p$  and Bob's report on  $q$ . In this way, agents are paid by a scoring function on a *correlated task* minus a regularized scoring function on two *uncorrelated tasks*.

■ **Algorithm 1**  $\Phi$ -divergence pairing mechanism with a scoring function  $K$  for two agents,  $\mathcal{M}^{\Phi, K}$ .

**Input:** A report profile  $(\hat{\mathbf{x}}, \hat{\mathbf{y}})$  where both Alice and Bob submit report for all  $m \geq 2$  tasks.

**Parameters:** A convex function  $\Phi : [0, \infty) \rightarrow \mathbb{R}$ , its conjugate  $\Phi^*$ , and a scoring function  $K : \mathcal{X} \times \mathcal{Y} \rightarrow \text{dom}(\Phi^*) \subseteq \mathbb{R}$ .

- 1: For Alice, arbitrarily pick three tasks  $b, p$  and  $q$  where  $p$  and  $q$  are distinct. We call  $b$  the *bonus task*,  $p$  the *penalty task to Alice*, and  $q$  the *penalty task to Bob*.
- 2: Based on Alice's reports on  $b$  and  $p$  ( $\hat{x}_b$  and  $\hat{x}_p$ ) and Bob's reports on  $b$  and  $q$  ( $\hat{y}_b$  and  $\hat{y}_q$ ), the payment to Alice is

$$M_A^{\Phi, K}(\hat{\mathbf{x}}, \hat{\mathbf{y}}) \triangleq K(\hat{x}_b, \hat{y}_b) - \Phi^*(K(\hat{x}_p, \hat{y}_q)). \quad (2)$$

- 3: The payment of Bob is defined similarly.

To simplify the notion, we use  $u_A$  or  $u_A(\boldsymbol{\theta}, P, K)$  to denote the ex-ante payment to Alice under a strategy profile  $\boldsymbol{\theta}$  and a joint signal distribution  $P$  in pairing mechanism with a scoring function  $K$ .

In general, the truthfulness guarantees of Mechanism 1 depends on the degeneracy of Alice's and Bob's signal distribution  $P$  and convex function  $\Phi$ . In this paper, we consider three different conditions which will be used in the statement of our results.

► **Assumption 10.** *In this paper, we consider the following three different settings.*

1. *no assumption;*
2.  *$P_{X,Y}$  is stochastic relevant;*
3. *Besides the above conditions,  $\mathcal{X}$  and  $\mathcal{Y}$  are finite sets,  $\Phi$  is strictly convex and differentiable, and  $\Phi^*$  is strictly convex.*

#### 3.1 Obtaining a Good Scoring Function

The  $\Phi$ -pairing mechanism  $\mathcal{M}^{\Phi, K}$  is not stand-alone mechanism for information elicitation, because it requires a scoring function  $K$  as a parameter. We will see shortly in Sect. 4 and 6, the truthfulness guarantees of the pairing mechanism depends on the quality of the scoring function. In this paper, we consider three different models for mechanism designers to estimate good scoring functions which are discussed in the rest of the sections:

**Direct access of  $K_{P, \Phi}^*$ .** In Sect. 4, we first consider the mechanism knows a  $(P, \Phi)$ -ideal scoring function  $K_{P, \Phi}^*$ . Note that if the mechanism knows the prior  $P$ , it can compute the  $(P, \Phi)$ -ideal scoring function, but the converse is not necessarily true.

**General reduction to a learning problem.** In Sect. 6, besides the reports from Alice and Bob, mechanism may exploit Alice and Bob's previous scoring function and other side information. For example the joint distribution between Alice and Bob can be approximated by some parametric model, say joint Gaussian distributions. We introduce our framework (Mechanism 2) that reduces the problem into a learning problem.

**Estimation from samples.** Finally, in the multi-task setting, if Alice and Bob truthfully report their signals, it is possible to estimate the  $(P, \Phi)$ -ideal scoring function from those reports. However, the mechanism needs to incentive them to be truthful. In Sect. 6.3, we propose two learning methods to estimate good scoring functions. Combining them with our framework (Mechanism 2), we can have detail-free  $\epsilon$ -strongly truthful mechanisms with high probability.

#### 4 Pairing Mechanisms in the Known Prior Setting

If the mechanism  $\mathcal{M}^{\Phi, K^*}$  has an  $(P, \Phi)$ -ideal scoring function  $K^*$  where  $P$  is the joint distribution to Alice's and Bob's signals, the mechanism has the following properties. We defer the proof to Sect. 5.

► **Theorem 11.** *Let an integer  $m$  be greater than 2, a functional  $\Phi$  be a continuous convex function with  $[0, \infty) \subseteq \text{dom}(\Phi)$ ,  $\mathbb{P}$  with  $P_{X,Y}$  be a common prior between Alice and Bob satisfying Assumption 2. Let  $\tau$  be the truth-telling strategy profile, and  $K^*$  be a  $(P, \Phi)$ -ideal scoring function.*

*The  $\Phi$ -pairing mechanism with  $K^*$ ,  $\mathcal{M}^{\Phi, K^*}$  has the following properties: For any strategy profile  $\theta$ ,<sup>8</sup>*

$$u_A(\theta, P, K^*) \leq u_A(\tau, P, K^*). \quad (3)$$

*Furthermore, under the four conditions in Assumption 10 respectively, the mechanism  $\mathcal{M}^{\Phi, K^*}$  is*

1. *truthful,*
2. *informed-truthful, or*
3. *strongly truthful.*

In the following example, we show how Mechanism 1 with a  $(P, \Phi)$ -ideal scoring function works, and illustrate the difference between informed-truthful and strongly truthful.

#### 5 Main Technical Lemmas and Proof of Theorem 11

To prove Theorem 11, we use the following lemmas which are also important in the rest of the paper.

We first show the ex-ante payment under the truth-telling strategy profile in the  $\Phi$ -pairing mechanism with  $(P, \Phi)$ -ideal scoring function is the  $\Phi$ -mutual information between Alice's and Bob's signals.

► **Lemma 12 (Truth-telling).** *If  $K^*$  is a  $(P_{X,Y}, \Phi)$ -ideal scoring function,*

$$u_A(\tau, P, K^*) = D_\Phi(P_{X,Y} \| P_X P_Y).$$

<sup>8</sup> There are some minor details when  $\mathcal{X}$  and  $\mathcal{Y}$  are not finite set. Here we require  $\theta$  to have finite  $\int K^* d\theta_A d\theta_B dP_{X,Y}$ , and  $\int \Phi^*(K^*) d\theta_A d\theta_B dP_X P_Y$ .



Moreover, if  $P_{X,Y}$  is stochastic relevant,  $D_\Phi(P_{X,Y} \| P_X P_Y) > 0$ .

Then we show any deviation from the truth-telling strategy profile or an ideal scoring function cannot improve Alice (and Bob's) ex-ante payment. The proof uses the variational representation of  $\Phi$ -divergence (Theorem 8).

► **Lemma 13** (Manipulation). *For any strategy profile  $\theta$  and scoring function  $K$ ,  $u_A(\theta, P, K) \leq D_\Phi(P_{X,Y} \| P_X P_Y)$ .*

Note that combining these two lemmas we have an even stronger result than inequality (3) which is a key tool in this paper: For any scoring function  $K$  and strategy profile  $\theta$ ,

$$u_A(\theta, P, K) \leq u_A(\tau, P, K^*). \quad (4)$$

► **Lemma 14** (Oblivious strategy). *If  $\theta$  is an oblivious strategy profile, for any scoring function  $K$ ,  $u_A(\theta, P, K) \leq 0$ .*

► **Lemma 15**. *Moreover, given Conditions 3 in Assumption 10, the equality in (4) for Alice or Bob occurs if and only if*

1.  $\theta = (\pi_A, \pi_B)$  which is a permutation strategy profile, and
2. For all  $x \in \mathcal{X}$  and  $y \in \mathcal{Y}$ ,  $K(\pi_A(x), \pi_B(y)) = \Phi'(\text{JP}(x, y))$ .

Informally, Lemma 15 shows if the pair of a strategy profile and a scoring function  $(\theta, K)$  have (4) equal only if there is a “conjugated” structure between the strategy and the scoring function. The proof uses the pigeonhole principle on the finite signal spaces and shows if the equality holds under a non permutation strategy profile,  $P$  is not stochastic relevant.

With the above four lemmas, we are ready to prove Theorem 11.

**Proof of Theorem 11.** There are four statements to show.

First, (3) is a direct result of (4). Furthermore, (3) proves that truth-telling is a Bayesian Nash equilibrium, and has highest ex-ante payment to Alice. This shows the mechanism is truthful.

By Lemma 14, the ex-ante payment to Alice (and Bob) is non-positive. Combining this and Lemma 12, we prove the  $\Phi$ -pairing mechanism with  $(P, \Phi)$ -ideal scoring function is informed-truthful when  $P$  is stochastic relevant.

To show our mechanism is strongly truthful, under Condition 3 in Assumption 10, we use the first part of Lemma 15. If the ex-ante payment under some strategy profile is equal to the ex-ante payment under the truth-telling strategy profile, the strategy profile is a permutation strategy profile. ◀

## 6 The Pairing Mechanism in the Detail Free Settings

With Sect. 5, we can see that to achieve the truthfulness guarantees, it suffices to have a “good” scoring function. That is if the ex-ante payment to Alice under the truth-telling strategy profile is close to the  $\Phi$ -mutual information between Alice's and Bob's signals, by (4), the ex-ante payment under an untruthful-strategy is less than the ex-ante payment under the truth-telling strategy profile.

In Sect. 6.1 we formalize the notions of a *good* scoring function and of the *accuracy* of a learning algorithm  $\mathcal{L}$  for scoring functions. In Sect. 6.2, we state our main result, Theorem 16, which reduces the mechanism design problem to a learning problem for an ideal scoring function, and provides some intuition about the proof of the theorem.

## 6.1 Accuracy of Scoring Rules and Learning Algorithms

Now we define a *good* scoring function, and the *accuracy* of a learning algorithm  $\mathcal{L}$ . Given  $\Phi$ , a prior  $P_{X,Y}$  and  $\epsilon > 0$ , we say that a scoring function  $K$  is  $\epsilon$ -ideal on  $(P_{X,Y}, \Phi)$ , if for Alice

$$u_A(\tau, P, K) \geq u_A(\tau, P, K_{P,\Phi}^*) - \epsilon = D_\Phi(P_{X,Y} \| P_X P_Y) - \epsilon, \quad (5)$$

and the similar inequality holds for Bob. Additionally, For  $m_L \in \mathbb{N}$ , we say a *learning algorithm for scoring functions with  $m_L$  samples*, as a function from  $(\mathbf{x}_L, \mathbf{y}_L) \in (\mathcal{X} \times \mathcal{Y})^{m_L}$  to a scoring function  $K$ . Given  $\mathcal{P}$ , a set of distributions on  $\mathcal{X} \times \mathcal{Y}$ , and a function  $S_L : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{N}$ , we say such a learning algorithm  $\mathcal{L}$  is  $(\delta, \epsilon)$ -accurate on  $(\mathcal{P}, \Phi)$  with  $S_L(\delta, \epsilon)$  samples, if for all  $P_{X,Y} \in \mathcal{P}$ ,  $\delta \in (0, 1)$ ,  $\epsilon > 0$ , and  $m_L \geq S_L(\delta, \epsilon)$ :

$$\Pr_{(\mathbf{x}_L, \mathbf{y}_L) \sim P_{X,Y}^{m_L}} [u_A(\tau, P, \mathcal{L}(\mathbf{x}_L, \mathbf{y}_L)) > D_\Phi(P_{X,Y} \| P_X P_Y) - \epsilon] \geq 1 - \delta.$$

That is, given  $m_L$  i.i.d. samples from  $P_{X,Y}$ , the probability that the output,  $\mathcal{L}(\mathbf{x}_L, \mathbf{y}_L)$ , is  $\epsilon$ -ideal on  $(P, \Phi)$  is greater than  $1 - \delta$ . Note that we require the algorithm  $\mathcal{L}$  to approximate the ideal scoring *uniformly* on all distributions in  $\mathcal{P}$ .

## 6.2 Pairing Mechanism with Learning Algorithms

Now we replace a fixed scoring function with an accurate learning algorithm  $\mathcal{L}$  in Mechanism 1. Intuitively, in the detail-free setting, the Mechanism 2 first runs a learning algorithm on Alice's and Bob's report profile to derive a scoring function, and then pays Alice and Bob by Mechanism 1.

■ **Algorithm 2**  $\Phi$ -divergence pairing mechanism with a learning algorithm  $\mathcal{M}^{\Phi, \mathcal{L}}$ .

**Parameters:** A convex function  $\Phi$ , and a learning algorithm  $\mathcal{L}$  with  $m_L$  samples.

**Input:** A report profile  $(\hat{\mathbf{x}}, \hat{\mathbf{y}})$  from Alice and Bob on  $m$  tasks where  $m \geq 2 + m_L$ .

- 1: Partition  $m$  tasks (arbitrarily) into a set of learning tasks  $M_L$  and a set of scoring tasks  $M_S$  where  $|M_L| \geq m_L$  and  $|M_S| \geq 2$ . Let  $(\hat{\mathbf{x}}_L, \hat{\mathbf{y}}_L)$  be the reports from Alice and Bob on the learning tasks  $M_L$ , and  $(\hat{\mathbf{x}}_S, \hat{\mathbf{y}}_S)$  be the reports on the scoring tasks.
- 2: Run the learning algorithm and derive  $K_{\text{est}} = \mathcal{L}(\hat{\mathbf{x}}_L, \hat{\mathbf{y}}_L)$ .
- 3: Run the  $\Phi$ -pairing mechanism (Mechanism 1) with the scoring function  $K_{\text{est}}$ , and pay Alice and Bob accordingly.

► **Theorem 16.** Let  $\Phi$  be a continuous convex function with  $[0, \infty) \subseteq \text{dom}(\Phi)$ ,  $m_L$  be an integer,  $\mathcal{L}$  be a learning algorithm on  $m_L$  samples, a function  $S_L : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{N}$ , and  $\mathcal{P}$  be a set of joint distributions on  $\mathcal{X} \times \mathcal{Y}$ .

Suppose the common prior between Alice and Bob satisfying Assumption 2 with  $P_{X,Y} \in \mathcal{P}$ , and  $\mathcal{L}$  is  $(\delta, \epsilon)$ -accurate on  $(\mathcal{P}, \Phi)$  with  $S_L(\delta, \epsilon)$  samples. Under three conditions in Assumption 10 respectively, Mechanism 2 is

1.  $(\delta, \epsilon)$ -truthful on  $\mathcal{P}$  with a  $2 + S_L(\delta, \epsilon)$  number of tasks;
2.  $(\delta, \epsilon)$ -informed-truthful on  $\mathcal{P}$  with a  $2 + S_L(\delta, \epsilon)$  number of tasks;
3.  $(\delta, \epsilon)$ -strongly truthful on  $\mathcal{P}$  with a  $2 + S_L(\delta, \epsilon)$  number of tasks.

Here  $\mathcal{L}$  only outputs an  $\epsilon$ -ideal scoring function on the joint distribution of agents' signals. Still, the algorithm can have an arbitrarily large error when agents are not truthtelling. For instance, there may exists a non-truth-telling strategy profile  $\theta$  such that  $\theta \circ P$  is not in  $\mathcal{P}$ ,

and the output of  $\mathcal{L}$  is not  $\epsilon$ -ideal on  $(\theta \circ P, \Phi)$ . Nevertheless, Mechanism 2 still can upper bound their ex-ante payment under such non-truth-telling strategy profiles. Furthermore, if the learning algorithm is  $\epsilon$ -ideal on  $(\theta \circ P, \Phi)$  for all strategy profile  $\theta$ , the pairing mechanism is indeed approximately dominantly truthful.

### 6.3 Learning Ideal Scoring Functions

Theorem 16 reduces the mechanism design problem to a learning problem for an ideal scoring function. However, Eqn. (5) may be hard to verify. We provide two natural sufficient conditions for  $\epsilon$ -ideal scoring functions in Sect. 6.3.1, and we will provide two concrete learning algorithms for scoring function in Sect. 6.3.2.

#### 6.3.1 Sufficient Conditions for Approximately $\Phi$ -Ideal Scoring Functions

**Bregman divergence.** Given  $a, b \in \mathbb{R}$  and a strictly convex and twice differentiable  $\Phi : \mathbb{R} \rightarrow \mathbb{R}$ , the standard Bregman divergence is  $\Phi(a) - \Phi(b) - \nabla \Phi(b)^\top (a - b)$ . It can be extended to **Bregman divergence** between two functionals  $f$  and  $g$  over a probability space  $(\Omega, \mathcal{F}, P)$  [4]

$$B_{\Phi, P}(f, g) = \int \Phi(f(\omega)) - \Phi(g(\omega)) - \nabla \Phi(g(\omega))^\top (f(\omega) - g(\omega)) dP(\omega).$$

► **Lemma 17** (Bregman divergence and accuracy). *If  $\Phi$  is strictly convex and twice differentiable on  $[0, \infty)$ ,  $D_\Phi(P_{X,Y} \| P_X P_Y) - u_A(\tau, P, K) = B_{\Phi^*, P_X P_Y}(K, K^*)$ . Therefore, if  $B_{\Phi^*, P_X P_Y}(K, K^*) \leq \epsilon$ ,  $K$  is an  $\epsilon$ -ideal scoring function on  $(\Phi, P)$ .*

Since Bregman divergence capture an *average distance* between a scoring function  $K$  and the ideal one, if the scoring function  $K$  is uniformly close to the ideal one  $K^*$ , the Bergman divergence between  $K$  and  $K^*$  is also small.

**Total variation distance.** On the other hand, we may first learn the prior  $P$  and compute an approximately ideal scoring function afterward. This indirect method is also useful, because estimating the probability density function is a much well studied problem.

► **Theorem 18** (Total variation to accuracy). *Given  $\Phi$  is a convex function and a prior  $P_{X,Y}$  over a finite space  $\mathcal{X} \times \mathcal{Y}$ , suppose there exist constants  $0 < \alpha < 1$  and  $c_L$  such that*

$$\forall x \in \mathcal{X}, y \in \mathcal{Y}, P_{X,Y}(x, y) > 2\alpha \text{ or } P_{X,Y}(x, y) = 0, \quad (6)$$

$$\forall z, w \in [\alpha, 1/\alpha], |\Phi(z) - \Phi(w)| \leq c_L |z - w|. \quad (7)$$

*If  $\|\hat{P}_{X,Y} - P_{X,Y}\|_{TV} \leq \delta < \alpha$ ,<sup>9</sup>  $\hat{K}(x, y) \in \partial \Phi \left( \frac{\hat{P}_{X,Y}}{\hat{P}_X \otimes \hat{P}_Y} \right)$  is a  $\frac{6c_L}{\alpha^2} \delta$ -ideal scoring function.*

The first condition says the smallest nonzero probability  $P_{X,Y}(x, y)$  is either constantly away from zero or equal to zero, and the second condition requires the function  $\Phi$  is Lipschitz in  $[\alpha, 1/\alpha]$ . With these conditions, if we have a good estimation  $\hat{P}$  for  $P$  with small total variation distance, we can compute a very accurate scoring function  $\hat{K}$  from  $\hat{P}$ . As we will see in Sect. 6.3.2, the empirical distributions with  $m_L$  samples satisfies this condition with high probability for large enough  $m_L$ .

<sup>9</sup>  $\|\hat{P} - \hat{P}\|_{TV} = \sum_{\omega \in \Omega} |P(\omega) - \hat{P}(\omega)|$  is the total variation distance between  $P$  and  $\hat{P}$ .

### 6.3.2 Learning Algorithms for Scoring Functions

**Generative approach.** Recall that if  $P$  is known, the ideal scoring function can be computed directly. In a generative approach, we try to estimate the probability density function  $P$  from reports and derive the scoring function afterward under the truth-telling strategy profile. In general this generative approach is useful when  $\mathcal{P}$  is on a finite space, or  $\mathcal{P}$  is a parametric model by Theorem 18. Here we provide an example of a generative approach.

A standard way of learning probability density function is to use empirical distribution on  $m_L$  samples. The following theorem shows that the empirical distribution gives a good estimation in terms of total variation distance.

► **Lemma 19** (Theorem 3.1 in [6]). *For all  $\epsilon > 0$ ,  $\delta > 0$ , finite domain  $\Omega$ , and distribution in  $P$  in  $\Delta_\Omega$ , there exists  $M = O\left(\frac{1}{\epsilon^2} \max(|\Omega|, \log(1/\delta))\right)$  such that for all  $m_L \geq M$  the empirical distribution  $\hat{P}_{m_L}$  with  $m_L$  i.i.d. samples,  $\|P - \hat{P}_{m_L}\|_{TV} \leq \epsilon$  with probability at least  $1 - \delta$ .*

Therefore, we can design a learning algorithm  $\mathcal{L}_{\text{emp}}$  as follows: estimate joint distribution  $P_{X,Y}$  by their empirical distributions  $\hat{P}_{X,Y}$  and derive  $\hat{K}$  from Theorem 18. By Theorem 18 and Lemma 19, such algorithm is  $\epsilon$ -accurate with  $1 - \delta$  probability.

**Discriminative approach.** Instead of density estimation, a discriminative approach estimates an ideal scoring functions directly. This enables more freedom of algorithm design. Here we use the variational representation (Theorem 8), and give an optimization characterization of an ideal scoring function.

Given the assumption 2, under the truth-telling strategy profile we can have i.i.d. samples of  $(u, v)$  where  $u$  is sampled from  $P_{X,Y}$  and  $v$  is sampled from  $P_X P_Y$  independently. Taking  $L^\Phi(a, b) \triangleq a - \Phi^*(b)$  as the risk function, we can convert the estimation of the ideal scoring functions to empirical risk minimization (maximization) over a training set  $(u_t, v_t)$  with  $t = 1, 2, \dots, \lfloor m_L/3 \rfloor$ ,

$$\tilde{K} = \arg \max_{k \in \mathcal{K}} \sum_t L^\Phi(k(u_t), k(v_t)) = \arg \max_{k \in \mathcal{K}} \left\{ \int k(\omega) d\hat{P}_{X,Y}(\omega) - \int \Phi^*(k(\omega)) dP_X \hat{P}_Y(\omega) \right\} \quad (8)$$

where  $\mathcal{K}$  is a pre-specified class of functionals  $k : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$ ,  $\hat{P}_{X,Y}$  and  $P_X \hat{P}_Y$  are empirical distributions on  $\lfloor m_L/3 \rfloor$  samples from distributions  $P_{X,Y}$  and  $P_X P_Y$  respectively.

Assuming that  $\mathcal{K}$  is a convex set of functionals, the implementation of (8) only requires solving a convex optimization problem over function space  $\mathcal{K}$  which is well studied [22]. With these results, we show the empirical risk maximizer  $\tilde{K}$  with respect to  $L^\Phi$  is  $\epsilon$ -accurate with large probability under some conditions on  $\mathcal{K}$  and prior  $P_{X,Y}$ . Furthermore, this error can be seen as the generalized error of the empirical risk maximizer.

► **Theorem 20.** *Consider a distribution  $P$  over  $\mathcal{X} \times \mathcal{Y}$ ; a strictly convex and a twice differentiable function  $\Phi$  on  $[0, \infty)$  with its gradient  $\Phi'$  and conjugate  $\Phi^*$ ; a family of functional  $\mathcal{K}$  from  $\mathcal{X} \times \mathcal{Y}$  to  $\text{dom}(\Phi^*)$ ; and  $\Phi^*(\mathcal{K}) = \{\Phi^*(k) : k \in \mathcal{K}\}$ . Suppose*

1. *the  $(P, \Phi)$ -ideal scoring function  $K^* = \Phi' \left( \frac{P_{X,Y}}{P_X P_Y} \right)$  is in  $\mathcal{K}$ , and*
2. *there exist constants  $(L_l, R_l, D_l)_{l=1,2}$* 
  - a.  $\sup_{k \in \mathcal{K}} \rho_{L_1}(k, P_{X,Y}) \leq R_1$ , and  $\int_0^{R_1} \sqrt{\mathcal{H}_{[\cdot], L_1}(u, \mathcal{K}, P_{X,Y})} du \leq D_1$
  - b.  $\sup_{l \in \Phi^*(\mathcal{K})} \rho_{L_2}(l, P_X P_Y) \leq R_2$  and  $\int_0^{R_2} \sqrt{\mathcal{H}_{[\cdot], L_2}(u, \Phi^*(\mathcal{K}), P_X P_Y)} du \leq D_2$

*There exists  $M = O\left(\frac{1}{\epsilon^2} \log \frac{1}{\delta}\right)$ , such that for all  $m_L \geq M$ ,  $\tilde{K}$  is  $\epsilon$ -accurate on prior  $P$  with probability  $1 - \delta$ .*

Informally, Theorem 20 requires the functional class  $\mathcal{K}$  contains an ideal scoring function and it has a constant complexity (generalized entropy with bracketing). Under these conditions, the empirical risk minimizer (maximizer) can estimate the ideal scoring function accurately even when the signal space can be integers, real numbers, or Euclidean spaces.

Here we give a outline of the proof. By Lemma 17, it is sufficient to show the empirical risk minimizer  $\tilde{K}$  has small Bregman divergence from the ideal one. Moreover, if the estimation  $K$  is the empirical risk maximizer, this error can be upper bounded by the distance between the empirical distribution and the real distribution (Lemma 21). Therefore, we can use functional form of Central Limit Theorem to upper bound the error. We defer the proof to the full version.

► **Lemma 21.** *Let  $\tilde{K}$  be the estimate of  $K^*$  obtained by solving Eqn. (8), and  $K^* \in \mathcal{K}$ . Then*

$$B_{\Phi^*, P_X P_Y}(\tilde{K}, K^*) \leq \sup_{k \in \mathcal{K}} \left| \int \Phi^*(k - \Phi^*(K^*)) d(\tilde{P}_X \tilde{P}_Y - P_X P_Y) - \int (k - K^*) d(\tilde{P}_{X,Y} - P_{X,Y}) \right|.$$

## 7 Machine Learning and Multiple Agents

We have discussed the  $\Phi$ -pairing mechanisms on two agents, Alice and Bob. What can we do if there are more than two agents, Alice, Bob, ...? We first discuss a naive approach that reduces the multiple agents setting to the two agent setting: Randomly select two agents, pay them according to a two agent mechanism, and pay the other agents 0. In this mechanism, as long as the mutual information between any two agents' signals is lower bounded, the sample complexity does not grow with the number of agents.

This naive approach is clearly wasteful and not useful in practice. It throws away nearly all the information agents provide, and will yield payments with high variance. Nonetheless it can provide some strong theoretical guarantees if the sole goal is obtaining approximately informed truthful mechanisms. In a way, this improves the sample complexity of Agarwal et al. [1] from  $O(n)$  to constant with an almost trivial analysis.

Yet, the progression of these papers does provide key insights. In Shnayder et al. [28], agents are paired up with every other agent, enough samples are drawn to estimate the pairwise joint distributions and this is used to (in our parlance) learn an ideal scoring function. Agarwal et al. [1] then assumes structure on the joint prior of all agents, and leverages this particular structure to better learn the ideal scoring function.

These approaches seem much more promising in practice. In the case where the mutual information between any two agents is lower bounded by a constant, they will not asymptotically improve the sample complexity, but in real life, constants matter.

Moreover, these technique may lead to better asymptotic analysis when the average pair-wise mutual information goes to zero. For example, say only Alice and Bob work on the tasks and the rest of agents report random noise. Alice will now only have positive expected payment if she and Bob are both randomly selected. As the number of agents increases, her expected payment will go to zero. However, if the sample complexity is large enough that a learning algorithm can pick out Alice and Bob from the crowd of noise, a mechanism might be able to appropriately reward them.

Intuitively, in such a case, we can pair Alice simultaneously with all other agents, and run our mechanism using the concatenation of all other agent's signals as "Bob"'s signal. As the number of agents increases, this approach ensures Alice's expected payment is non-decreasing because the mutual information does not decrease by adding more information – the additional agents' reports. However, the sample complexity for ideal scoring function will increase, perhaps even exponentially.

We propose two novel approaches that exploit the power of current machine learning algorithms to compute the mutual information between Alice’s signal and the rest of the agents with limited sample complexity.

**Computing the  $\Phi$ -Mutual Information between  $X_i$  and  $X_{-i}$ .** Our variation method is well suited to the challenge of reliably computing the  $\Phi$ -mutual information between Alice’s reports,  $X_i$ , and those of the other agents,  $X_{-i}$ .

Recall that Mechanism 2 reduces the mechanism design problem to learning a scoring rule, which Eqn. (1) reduces to learning

$$\text{JP}_P(x_i, x_{-i}) = \frac{P_{X_i, X_{-i}}(x_i, x_{-i})}{P_{X_i}(x_i)P_Y(x_{-i})} = \frac{P_{X_i|X_{-i}}(x_i | x_{-i})}{P_{X_i}(x_i)}.$$

Therefore, it is enough to learn both the marginal distribution,  $P_{X_i}(x_i)$  and  $P_{X_i|X_{-i}}(x_i | x_{-i})$ . The former can be estimated empirically. However, when the number of agents is large, the later is high dimensional and must be learned. Fortunately, this is just a soft-classifier which produces a forecast to predict her report rather than a single report. which, given the reports of every agent but Alice on a particular task, (soft) predicts Alice’s report on the same task.

Therefore, we can derive an approximate ideal scoring rule by using machine learning techniques to produce a (soft) prediction of Alice’s report for an answer given the reports of the other agents. Specifically, the machine learning algorithm outputs  $f(\cdot, \cdot)$  such that  $f(x_i, x_{-i}) = P_{X_i|X_{-i}}(x_i | x_{-i})$ .

Using Mechanism 2, we can divide the tasks into training and testing tasks. The training tasks are used to learn  $f$  and to estimate  $P_X(x)$ . We can compute  $K_{\text{est}}$  from  $f$  and  $P_X(x)$ , and then use Mechanism 2 to pay the agents.

Note that for the guarantees of Theorem 16 to hold, it is required that  $f$  is learned accurately on truthful strategy profiles. However, we do not require the learning algorithms perform well on non-truthful strategy profiles.

**Latent Variable Models.** Our pairing mechanisms are particularly powerful when the prior  $P$  on agents’ signals is a latent variable model. In a latent variable model, signals are mutually independent conditioned on the latent variables. Examples include Dawid-Skene models, Gaussian mixture models, hidden Markov models, and latent Dirichlet allocations. When  $P$  is a latent variable model, we can pay Alice the (approximate) mutual information between her report and each task’s latent variable.

1. Given a latent label recovery algorithm, e.g., [33], we run such algorithm on all reports except Alice’s, and get estimate of latent label for each tasks  $(Y_1, \dots, Y_m)$ ;
2. Then, using Alice’s report  $(X_1, \dots, X_m)$  and the estimated latent label  $(Y_1, \dots, Y_m)$  we can run Mechanism 2 with the generated method to pay Alice the mutual information between Alice’s report and the latent labels.

This mechanism is (approximate) strongly truthful, because the  $\Phi$ -mutual information between Alice and the others’ reports is less than the  $\Phi$ -mutual information between her reports and the tasks’ latent variable due to data processing inequality. This approach has the following advantages. First, this provides a reduction from aggregation to elicitation. Second, paying mutual information between Alice’s reports and the latent variable resolves the problems that the above naive approaches have. Alice’s payment increases as the number of agents increases by the data processing inequality and the sample complexity of scoring function mirrors that of the latent label algorithm, which typically will not increase.



## References

- 1 Arpit Agarwal, Debmalya Mandal, David C Parkes, and Nisarg Shah. Peer prediction with heterogeneous users. In *Proceedings of the 2017 ACM Conference on Economics and Computation*, pages 81–98. ACM, June 2017.
- 2 Syed Mumtaz Ali and Samuel D Silvey. A general class of coefficients of divergence of one distribution from another. *Journal of the Royal Statistical Society: Series B (Methodological)*, 28(1):131–142, 1966.
- 3 Imre Csiszár. Eine informationstheoretische ungleichung und ihre anwendung auf beweis der ergodizitaet von markoffschen ketten. *Magyer Tud. Akad. Mat. Kutato Int. Koezl.*, 8:85–108, 1964.
- 4 Imre Csiszár. Generalized projections for non-negative functions. *Acta Mathematica Hungarica*, 68(1-2):161–186, 1995.
- 5 Anirban Dasgupta and Arpita Ghosh. Crowdsourced judgement elicitation with endogenous proficiency. In *Proceedings of the 22nd international conference on World Wide Web*, pages 319–330. ACM, 2013.
- 6 Luc Devroye and Gábor Lugosi. *Combinatorial methods in density estimation*. Springer Science & Business Media, 2012.
- 7 Boi Faltings and Goran Radanovic. Game theory for data science: eliciting truthful information. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 11(2):1–151, 2017.
- 8 Werner Fenchel. On conjugate convex functions. *Canadian Journal of Mathematics*, 1(1):73–77, 1949.
- 9 Alice Gao, James R Wright, and Kevin Leyton-Brown. Incentivizing evaluation via limited access to ground truth: Peer-prediction makes things worse. *Workshop on Algorithmic Game Theory and Data Science at ACM Conference on Economics and Computation*, 2016.
- 10 Naman Goel and Boi Faltings. Personalized peer truth serum for eliciting multi-attribute personal data. In *UAI*, 2019.
- 11 Yuqing Kong. Dominantly truthful multi-task peer prediction with a constant number of tasks. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2398–2411. SIAM, 2020.
- 12 Yuqing Kong, Katrina Ligett, and Grant Schoenebeck. Putting peer prediction under the micro (economic) scope and making truth-telling focal. In *International Conference on Web and Internet Economics*, pages 251–264. Springer, 2016.
- 13 Yuqing Kong and Grant Schoenebeck. Eliciting expertise without verification. In *Proceedings of the 2018 ACM Conference on Economics and Computation*, pages 195–212. ACM, 2018.
- 14 Yuqing Kong and Grant Schoenebeck. Equilibrium selection in information elicitation without verification via information monotonicity. In *9th Innovations in Theoretical Computer Science Conference*, 2018.
- 15 Yuqing Kong and Grant Schoenebeck. Water from two rocks: Maximizing the mutual information. In *Proceedings of the 2018 ACM Conference on Economics and Computation*, pages 177–194. ACM, 2018.
- 16 Yuqing Kong and Grant Schoenebeck. An information theoretic framework for designing information elicitation mechanisms that reward truth-telling. *ACM Transactions on Economics and Computation (TEAC)*, 7(1):2, 2019.
- 17 Yuqing Kong, Grant Schoenebeck, Biaoshuai Tao, and Fang-Yi Yu. Information elicitation mechanisms for statistical estimation. In *AAAI*, pages 2095–2102, 2020.
- 18 Yang Liu and Yiling Chen. Machine-learning aided peer prediction. In *Proceedings of the 2017 ACM Conference on Economics and Computation*, EC '17, pages 63–80, New York, NY, USA, 2017. ACM. doi:10.1145/3033274.3085126.
- 19 Yang Liu and Yiling Chen. Surrogate scoring rules and a dominant truth serum for information elicitation. *CoRR*, abs/1802.09158, 2018. arXiv:1802.09158.
- 20 N. Miller, P. Resnick, and R. Zeckhauser. Eliciting informative feedback: The peer-prediction method. *Management Science*, pages 1359–1373, 2005.



- 21    Tetsuzo Morimoto. Markov processes and the h-theorem. *Journal of the Physical Society of Japan*, 18(3):328–331, 1963. doi:10.1143/JPSJ.18.328.
- 22    XuanLong Nguyen, Martin J Wainwright, and Michael I Jordan. Estimating divergence functionals and the likelihood ratio by convex risk minimization. *IEEE Transactions on Information Theory*, 56(11):5847–5861, 2010.
- 23    D. Prelec. A Bayesian Truth Serum for subjective data. *Science*, 306(5695):462–466, 2004.
- 24    Goran Radanovic and Boi Faltings. A robust bayesian truth serum for non-binary signals. In Marie desJardins and Michael L. Littman, editors, *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence, July 14-18, 2013, Bellevue, Washington, USA*. AAAI Press, 2013. URL: <http://www.aaai.org/ocs/index.php/AAAI/AAAI13/paper/view/6451>.
- 25    Goran Radanovic and Boi Faltings. Incentives for truthful information elicitation of continuous signals. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.
- 26    Ralph Tyrell Rockafellar. *Convex analysis*. Princeton university press, 2015.
- 27    Grant Schoenebeck and Fang-Yi Yu. Two strongly truthful mechanisms for three heterogeneous agents answering one question. In *International Conference on Web and Internet Economics*. Springer, 2020.
- 28    Victor Shnayder, Arpit Agarwal, Rafael Frongillo, and David C Parkes. Informed truthfulness in Multi-Task peer prediction. In *Proceedings of the 2016 ACM Conference on Economics and Computation*, EC ’16, pages 179–196, New York, NY, USA, 2016. ACM.
- 29    Sara A van de Geer and Sara van de Geer. *Empirical Processes in M-estimation*, volume 6. Cambridge university press, 2000.
- 30    Jon Wellner et al. *Weak convergence and empirical processes: with applications to statistics*. Springer Science & Business Media, 2013.
- 31    Jens Witkowski and David C Parkes. Peer prediction without a common prior. In *Proceedings of the 13th ACM Conference on Electronic Commerce*, pages 964–981. ACM, 2012.
- 32    Peter Zhang and Yiling Chen. Elicitability and knowledge-free elicitation with peer prediction. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, pages 245–252, 2014.
- 33    Yuchen Zhang, Xi Chen, Dengyong Zhou, and Michael I. Jordan. Spectral methods meet EM: A provably optimal algorithm for crowdsourcing. *J. Mach. Learn. Res.*, 17:102:1–102:44, 2016. URL: <http://jmlr.org/papers/v17/14-511.html>.

# Distributed Load Balancing: A New Framework and Improved Guarantees

**Sara Ahmadian**

Google Research, New York, NY, USA  
sahmadian@google.com

**Allen Liu**

MIT, Cambridge, MA, USA  
cliu568@mit.edu

**Binghui Peng**

Columbia University, New York, NY, USA  
bp2601@columbia.edu

**Morteza Zadimoghaddam**

Google Research, Cambridge, MA, USA  
zadim@google.com

---

## Abstract

Inspired by applications on search engines and web servers, we consider a load balancing problem with a general *convex* objective function. In this problem, we are given a bipartite graph on a set of sources  $S$  and a set of workers  $W$  and the goal is to distribute the load from each source among its neighboring workers such that the total load of workers are as balanced as possible. We present a new distributed algorithm that works with *any* symmetric non-decreasing convex function for evaluating the balancedness of the workers' load. Our algorithm computes a nearly optimal allocation of loads in  $O(\log n \log^2 d / \epsilon^3)$  rounds where  $n$  is the number of nodes,  $d$  is the maximum degree, and  $\epsilon$  is the desired precision. If the objective is to minimize the maximum load, we modify the algorithm to obtain a nearly optimal solution in  $O(\log n \log d / \epsilon^2)$  rounds. This improves a line of algorithms that require a polynomial number of rounds in  $n$  and  $d$  and appear to encounter a fundamental barrier that prevents them from obtaining poly-logarithmic runtime [6, 7, 13, 15]. In our paper, we introduce a novel primal-dual approach with multiplicative weight updates that allows us to circumvent this barrier. Our algorithm is inspired by [1] and other distributed algorithms for optimizing linear objectives but introduces several new twists to deal with general convex objectives.

**2012 ACM Subject Classification** Theory of computation → Distributed algorithms

**Keywords and phrases** Load balancing, Distributed algorithms

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.79

**Funding** *Binghui Peng*: Research supported by NSF IIS-1838154, NSF CCF-1703925 and NSF CCF-1763970.

## 1 Introduction

Emerging web based services including commercial web search engines face challenging resource efficiency targets in their serving data centers. Motivated by the growing demand in fast responding services, they aim for sub-second latency targets. Therefore they replicate the data across distributed machines in data centers to allow for serving queries in parallel as well as cloning the search algorithm to expedite computation tasks. With billions of queries to serve on a daily basis [14], load balancing becomes a critical challenge in resource efficiency and optimizing the computation fleet.



© Sara Ahmadian, Allen Liu, Binghui Peng, and Morteza Zadimoghaddam;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 79; pp. 79:1–79:20



Leibniz International Proceedings in Informatics

LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

From the combinatorial optimization perspective, one can formulate the serving requirements with packing and covering constraints and model this problem as an allocation/matching instance in a bipartite graph. Although feasibility of the allocation is the first problem to study, in practice, we face a wider range of objectives to optimize. Emergency mechanisms in data centers allow the excess load to be served with the buffer capacities locally or be shifted to alternative data centers globally with services like Global Server Load Balancing.

This motivates service level objectives (SLO) in terms of quantile statistics of machine utilization values or other convex functions that are much more sensitive to higher utilization values instead of standard linear objective functions in matching theory that have uniform partial derivatives across the whole range of valid utilization values. One particular frequently occurring scenario is when the underlying properties of the load balancing instance determines some phase transition utilization threshold (say 0.95) beyond which the service starts to deteriorate. Thus, the cost objective function we are trying to minimize should have completely different behaviours on the two sides of this threshold and linear functions are unable to capture this exponential growth in the cost function.

To accommodate this wide range of objectives, in this paper we focus on the load balancing problem with a general *convex* objective function. The problem is defined on a bipartite graph, with a set of sources  $S$  on one side, and a set of workers  $W$  on the other. For each source, we must distribute its load among its neighboring workers. The goal is to minimize a convex function of the workers' loads (where the load of a worker is the total load it receives from all incoming sources). Fractional allocations are allowed, as in the real-world setting, there are usually huge amounts of query requests coming from each source.

The problem described above can be solved as a convex program. However, in real-world settings, the graph could contain billions of nodes and be too large to store in one piece. Thus, for an algorithm to be scalable, it has to be implementable in a *distributed* manner. In this paper, we work in the CONGEST model that is standard in distributed algorithms literature (see e.g. [4]). Under this model, computation proceeds in rounds and in each round, each node may send a logarithmic number of bits to each of its neighbors. Previous distributed algorithms for load balancing (with a convex objective), such as [6, 7, 13, 15], require a number of distributed rounds that is *polynomial* in the number of nodes or the maximum degree of a node. This could still be prohibitive for real-world applications where each node could have a large number of neighbors. The main contribution of this paper is to provide *the first distributed algorithm* that computes an approximately optimal solution to the load balancing problem with a convex objective and runs in a *poly-logarithmic* number of distributed rounds. Our main theorem is stated below.

► **Theorem 1 (Informal).** *Assume the objective function  $\Phi$  is convex, symmetric, and non-decreasing in each variable. Then for any  $\epsilon < 1$ , our algorithm computes a  $(1 + \epsilon)$ -approximation to the optimal solution in  $O\left(\frac{\log n \log^2 d}{\epsilon^3}\right)$  rounds, where  $n, d$  denote the number of nodes and the maximum degree respectively.*<sup>1</sup>

In the special case where the objective is the max function, i.e. the goal is to minimize the maximum load, we obtain a distributed algorithm with an improved round complexity of  $O\left(\frac{\log n \log d}{\epsilon^2}\right)$ , which is faster than a direct application of a parallel mixed positive LP solver (see [11]) by an  $O\left(\frac{\log n}{\epsilon}\right)$  factor.

---

<sup>1</sup> Technically, we need a few additional assumptions on the objective function  $\Phi$  in order to obtain a  $(1 + \epsilon)$ -approximation in terms of objective value. Formally, our theorem is stated in terms of an  $\epsilon$ -approximate solution which is defined in the main body but for natural objective functions such as  $L^p$  norms for  $p > 1$ , our algorithm obtains a  $(1 + \epsilon)$ -approximation in objective value. See the remark at the end of Section 2 for more details.

► **Theorem 2** (Informal). *When the objective is the max function, our algorithm computes a  $(1 + \epsilon)$ -approximation to the optimal solution and runs in  $O\left(\frac{\log n \log d}{\epsilon^2}\right)$  rounds.*

## 1.1 Related work

There is extensive work on distributed algorithms for optimizing linear objective functions such as packing, covering and positive LPs [2, 3, 5, 10, 11, 17, 18]. In these settings, distributed algorithms with poly-logarithmic convergence rates are known. In particular, for general positive LPs, there is a distributed algorithm that computes a  $(1 + \epsilon)$ -approximation in  $O\left(\frac{\log^3 n}{\epsilon^3}\right)$  rounds; for the special case of pure packing and pure covering LPs, a better runtime of  $O\left(\frac{\log^2 n}{\epsilon^2}\right)$  is known [11]. We refer the interested reader to the thesis [16] for a more detailed survey. While the techniques in these works can be applied to our problem when the objective function is linear, optimizing a general convex objective is significantly different. Whereas for a linear objective, the optimum always occurs at one of the vertices of the feasible polytope, for a convex objective, the optimum may be in the interior. Therefore, it seems that existing algorithms for optimizing linear objectives cannot be directly applied.

As mentioned previously, there are works that study a convex load balancing objective [6, 7, 13, 15]. The algorithms proposed in these works have distributed runtime that depends polynomially on the number of nodes or the maximum degree whereas our algorithm has only polylogarithmic dependence on these parameters.

There has also been work on discrete load balancing i.e. when the sources are not divisible (so fractional allocations are not allowed). A recent line of work [4, 8, 9] obtains constant-factor approximation algorithms in the local and congest models. More specifically, Czygrinow et al. [8] give a distributed 2-approximation algorithm that runs in  $O(d^5)$  rounds. Assadi et al. [4] gives an  $O(1)$ -approximation algorithm for unweighted loads and an  $O(\log n)$  approximation algorithm for weighted loads with polylogarithmic round complexity in the congest model. For the less restrictive local model, they present an  $O(1)$ -approximation algorithm for weighted loads with polylogarithmic round complexity. In general, when the sources are not divisible, one cannot hope to compute, say, a  $(1 + \epsilon)$ -approximation efficiently. Thus, these works focus on constant-factor approximation, whereas in our setting, we focus on computing a nearly optimal solution.

## 1.2 Technical Overview

One key limitation for previous works on (continuous) load balancing [6, 7, 13, 15] is that they rely on an algorithm that, for each source, additively shifts load from higher-loaded neighbors to lower-loaded neighbors. However, the step size must be set to  $O(1/d)$  in order for the algorithm to be stable and thus the number of rounds required depends linearly (or even worse) on the maximum degree  $d$ .

Since the load balancing problem is convex, another natural approach is to apply general convex optimization algorithms that can be implemented in a distributed manner. However, straight-forward applications of first-order convex optimization algorithms also get stuck with a linear dependence on  $d$  because both the diameter of the feasible polytope and the condition number of the convex optimization problem can depend linearly on the maximum degree.

We circumvent the aforementioned limitations by adopting a different algorithmic framework and we introduce a novel algorithm based on a primal-dual approach with multiplicative-weight updates. One of the central insights in our algorithm is that if we know the target

capacities in the optimal solution, then we can compute the allocation of the sources that achieves the optimum as this essentially reduces to solving a linear problem. More generally, for a given set of target capacities, we can essentially test whether it is achievable. This motivates the following iterative procedure: we start with very high target capacities and iteratively update the target capacities downwards while checking feasibility until we reach a solution that is barely feasible.

For checking feasibility, we use a proportional allocation algorithm based on the work in [1] for maximum matching. We update the load assignment according to the proportional allocation algorithm and then update the target capacities based on whether each worker has too much or too little load. We start our algorithm with a feasible solution (by setting the target capacities to be very high) and can easily maintain feasibility throughout the entire process. The difficulty lies in proving optimality. An important observation is that after running the proportional allocation algorithm for a sufficient number of iterations, if there are workers whose load differs significantly from their target capacity, then it is possible to (implicitly) construct a certificate that this imbalance must happen in any feasible allocation. For the special case when the objective is the max function, it is not too difficult to complete the proof using the above ideas as it suffices to maintain the same target capacity for all the workers and decrease all capacities together. When the algorithm stops, we only need to show that there exists a set of workers that is *saturated*, in the sense that their total capacity is roughly equal to the total load of the sources whose neighbors are all included in this set. For general convex objectives, we will need a more refined analysis since the workers may have different target capacities. We will prove that the solution that we compute is optimal at multiple levels. In particular, our algorithm allows us to (implicitly) construct a multi-level cut that serves as a hierarchy of certificates that, when combined, imply that the entire solution is nearly optimal.

## 2 Preliminary

We now formalize our problem and introduce notation. In the load balancing problem, the input is a bipartite graph  $G(S, W, E)$ , where we use  $S$  ( $|S| = n_S$ ) to denote the set of sources and  $W$  ( $|W| = n_W$ ) to denote the set of workers. We write  $n = n_S + n_W$ . We assume there is a load associated with each source  $s \in S$ . For simplicity of presentation, we assume each source has one unit of load, although following the same proof method, the results obtained in this paper can be generalized to arbitrary weighted loads. For each source  $s \in S$ , we use  $N_s$  to denote the set of workers that are connected to the source  $s$ . Similarly, we use  $N_w$  to denote all the sources that are connected to worker  $w$ . We use  $d$  to denote the maximum degree of a node (source or worker) in the graph. For a subset of workers  $X \subset W$ , let  $N(X) \subset S$  be the set of sources  $s$  with the property that all of the neighbors of  $s$  are in  $X$ .

We want to assign loads along the edges of the graph such that all of the sources are served and the loads of the workers are as balanced as possible. We use  $x_{s,w}$  to denote the amount of load assigned from source  $s$  to worker  $w$ . We assume the load is splittable and fractional allocations are allowed. For each worker  $w$ , define its load  $L_w$  to be

$$L_w = \sum_{s \in N_w} x_{s,w}.$$

In this paper, our objective is to minimize some convex function of the loads. We assume the objective function  $\Phi : \mathbb{R}^{n_W} \rightarrow \mathbb{R}$  is symmetric, convex, and non-decreasing in each variable. Formally, we aim to solve the following optimization problem in a distributed manner:

$$\begin{aligned}
& \text{minimize} && \Phi(L_1, \dots, L_{n_W}) \\
& \text{subject to} && \sum_{w \in N_s} x_{s,w} = 1 \quad \forall s \in S \\
& && L_w = \sum_{s \in N_w} x_{s,w} \quad \forall w \in W \\
& && x_{s,w} \geq 0 \quad \forall s \in S, w \in W \\
& && x_{s,w} = 0 \quad \forall (s, w) \notin E
\end{aligned} \tag{1}$$

Given any load vector  $L = (L_1, \dots, L_{n_W}) \in \mathbb{R}^{n_W}$ , we say it is *feasible* if there exists a feasible assignment  $\{x_{s,w}\}_{s \in S, w \in W}$  that satisfies the constraints of Eq. (1). Our algorithm will find an  $\epsilon$ -approximate solution as defined below.

► **Definition 3** ( $\epsilon$ -approximate solution). *We say  $L = (L_1, \dots, L_{n_W})$  is an  $\epsilon$ -approximate solution, if  $L$  is feasible and for any other feasible solution  $(L'_1, \dots, L'_{n_W})$ , we have*

$$\Phi(L_1, \dots, L_{n_W}) \leq \Phi((1 + \epsilon)L'_1, \dots, (1 + \epsilon)L'_{n_W}).$$

We are primarily interested in the case where  $1/\epsilon$  is constant or poly-logarithmic in  $n$  so runtime that is polynomial in  $1/\epsilon$  is acceptable.

► **Remark 4.** The definition of  $\epsilon$ -approximate solution is stated in terms of scaling the *inputs* of the convex function rather than scaling the *value* of the convex function itself. An  $\epsilon$ -approximate solution would directly imply a  $(1 + O(\epsilon))$  approximation on the optimal value for any Lipschitz continuous function together with a mild lower bound assumption on the optimal value. In particular, it implies  $(1 + \epsilon)$ -approximation (in terms of objective value) for widely used functions such as  $L_p$  norms for  $p \geq 1$ .

In general, it is impossible to achieve a multiplicative approximation without additional assumptions on the objective. For instance, if we take  $\Phi = \max(\max(L_1, \dots, L_{n_W}) - \text{OPT}, 0)$  where  $\text{OPT}$  is the minimum possible value of  $\max(L_1, \dots, L_{n_W})$ , then achieving a multiplicative approximation would require solving the problem exactly.

### 3 Algorithm

We first give a high level overview of our main algorithm (the pseudocode is given in Algorithm 1). The algorithm maintains a target capacity  $C_w$  and a weight  $a_w$  for each worker  $w \in W$ . For any source  $s \in S$ , the amount of load it assigns to its neighbor worker  $w$  ( $w \in N_w$ ) is proportional to the weight  $a_w$  (see Line 13). Our algorithm always maintains a (near) feasible solution under the target capacity. It gradually decreases the target capacity and adjusts the weights, until reaching a nearly optimal solution. The algorithm updates the capacity once per epoch (see Line 8 to Line 12). While in each epoch, the algorithm freezes the target capacity and attempts to find a feasible assignment via the PROPORTIONAL ALLOCATION algorithm (see Algorithm 2). The PROPORTIONAL ALLOCATION algorithm was originally proposed in [1] for finding an approximate maximum matching. It adjusts the weight of workers based on the following rules: If the current load exceeds the target capacity, it decreases the weight by a multiplicative factor of  $(1 + \epsilon)$ ; it increases the weight otherwise. After repeating the weight updating rules for  $\tilde{O}\left(\frac{\log n \log d}{\epsilon^2}\right)$  steps, our algorithm updates the target capacities on workers based on the gap between the target capacity and the current load. The algorithm will *fix* workers whose load significantly exceeds the target capacity meaning that for these workers, the target capacity never changes anymore.

---

**Algorithm 1** GENERAL LOAD BALANCING.

---

```

1: Input: Graph  $G$ , set of workers  $W$ , set of sources  $S$ 
2: Initialize weights  $a_w = 1$  for all workers  $w \in W$ .
3: Initialize capacity upper bounds  $C_w = d$  for all workers  $w \in W$ 
4: Set all workers  $w$  to be unfixed
5: Set  $A = \frac{2 \log(d/\epsilon)}{\epsilon}$  and  $B = \frac{100 \log(n/\epsilon) \cdot \log(d/\epsilon)}{\epsilon^2}$ 
6: for  $r = 0, 1, \dots, A - 1$  do
7:    $x \leftarrow \text{PROPORTIONAL\_ALLOCATION}(G, C, a, \epsilon, B)$ 
8:   for  $w \in W$  do
9:     if  $w$  is unfixed and  $L_w < (1 + 10\epsilon)C_w$  then
10:      Set  $C_w \leftarrow \frac{C_w}{1+\epsilon}$ 
11:     else
12:      Set  $w$  to fixed
13: return  $x_{s,w} = \frac{a_w}{\sum_{w \in N_s} a_w}$ 

```

---

The pseudocode of the PROPORTIONAL ALLOCATION algorithm is given in Algorithm 2. We note that each time we run PROPORTIONAL ALLOCATION and update the weights  $a_w$ , we *do not* reinitialize the weights. We continue updating from the weights computed in the previous step.

---

**Algorithm 2** PROPORTIONAL ALLOCATION [1] ( $G, C, a, \epsilon, B$ ).

---

```

Input: Graph  $G$ , set of workers  $W$ , set of sources  $S$ 
Input: Target capacity  $C_w$  for each worker
Input: Initial weight  $a_w$  for each worker
Input: Precision  $\epsilon$ 
input: Number of rounds  $B$ 
for  $t = 0, 1, \dots, B - 1$  do
  Set  $x_{s,w} = \frac{a_w}{\sum_{w \in N_s} a_w}$  for all edge variables  $x_{s,w}$ 
  for  $w \in W$  do
    If  $L_w > C_w$  then update  $a_w \leftarrow \frac{a_w}{1+\epsilon}$ 
    If  $L_w < C_w$  then update  $a_w \leftarrow (1 + \epsilon)a_w$ 
return  $x_{s,w} = \frac{a_w}{\sum_{w \in N_s} a_w}$ 

```

---

## 4 Analysis

Our main result is formally stated in Theorem 5, we sketch the high-level idea of the proof here. The main idea in the proof of Theorem 5 is to show that if we have fixed the sets of workers  $W_1, W_2, \dots, W_r$  in iterations  $1, 2, \dots, r$ , then the number of sources whose only neighbors are in the set  $W_1 \cup \dots \cup W_r$  is at least  $(1 - O(\epsilon)) \sum_{w \in W_1 \cup \dots \cup W_r} C_w$ . This would then certify that our solution is essentially optimal for the total load among the set of workers in  $W_1 \cup \dots \cup W_r$ . While this claim is not technically true, Lemma 12 is a slight modification that involves considering a superset of  $W_1 \cup \dots \cup W_r$  and it suffices for our purposes. Once we have a hierarchy of certificates for  $r = 1, 2, \dots, A - 1$  we can prove that the solution computed by our algorithm is essentially optimal overall.



► **Theorem 5.** Assume the objective function  $\Phi : \mathbb{R}^{nw} \rightarrow \mathbb{R}^+$  is convex, symmetric, and non-decreasing in each variable. For  $0 < \epsilon < 1$ , Algorithm 1 computes an  $O(\epsilon)$ -approximate solution in  $O\left(\frac{\log n \log^2 d}{\epsilon^3}\right)$  distributed rounds.

In the proof of Theorem 5, the bulk of the work is in proving Lemma 12. We will first prove some basic facts about the behavior of the loads and target capacities throughout the execution of Algorithm 1 in Section 4.1. We then introduce the concept of majorization for analyzing convex functions in Section 4.2. In Section 4.3, we prove Lemma 12. A key observation about the proportional allocation algorithm of [1] is that at the end, if the load on a worker is significantly less than the target capacity, its weight  $a_w$  must be increased at every round and if the load on a worker is significantly more than the target capacity, its weight  $a_w$  must be decreased at every round. Thus, there must be a large multiplicative gap between weights on underallocated and overallocated workers. Since loads are allocated proportionally, if some source is connected to both underallocated workers and overallocated workers, almost all of the load is actually being sent to the underallocated workers. Exploiting this intuition (we will need a slightly more precise statement in the proof), we can construct the desired certificate and complete the proof of Lemma 12. Finally, combining Lemma 12 with the tools introduced in Section 4.2 for analyzing convex functions, we complete the proof of Theorem 5.

## 4.1 Basic Observations

### Notation

For a load variable  $L_w$  and indices  $0 \leq r < A, 0 \leq t \leq B$ , we let  $L_w^{r,t}$  denote its value when the algorithm is executed to the timestep  $r, t$  i.e. we have completed  $r$  full iterations of PROPORTIONAL ALLOCATION and  $t$  rounds within the next iteration of PROPORTIONAL ALLOCATION. We adopt the same notation for  $x_{s,w}$ ,  $s_w$  and  $C_w$ .

Below is an informal summary of the properties that we will prove in this section.

- Loads gradually move toward the target capacities
- Loads never significantly exceed the target capacity
- For fixed workers, their load is roughly equal to their target capacity
- Any significantly underallocated workers must have their weight increased at every previous timestep

We begin with a simple observation that after each weight update, the load on each worker moves toward the target capacity.

► **Lemma 6.** Consider indices  $0 \leq r < A, 0 \leq t < B$  then

- If  $L_w^{r,t} > C_w^{r,t}$  then  $\frac{L_w^{r,t}}{(1+\epsilon)^2} \leq L_w^{r,t+1} \leq L_w^{r,t}$
- If  $L_w^{r,t} < C_w^{r,t}$  then  $L_w^{r,t} \leq L_w^{r,t+1} \leq (1+\epsilon)^2 L_w^{r,t}$

**Proof.** We prove the first claim and the second one follows from the same argument. Suppose  $L_w^{r,t} > C_w^{r,t}$ , then we know that  $a_w^{r,t+1} = a_w^{r,t}/(1+\epsilon)$ . Together with the fact that  $a_{w'}^{r,t} \in [(1+\epsilon)^{-1}a_w^{r,t}, (1+\epsilon)a_w^{r,t}]$  holds for all worker  $w' \in W$ , we have

$$L_w^{r,t+1} = \sum_{s \in N_w} \frac{a_w^{r,t+1}}{\sum_{w' \in N_s} a_{w'}^{r,t+1}} \leq \sum_{s \in N_w} \frac{a_w^{r,t} \cdot (1+\epsilon)^{-1}}{\sum_{w' \in N_s} a_{w'}^{r,t} \cdot (1+\epsilon)^{-1}} = L_w^{r,t}$$

and

$$L_w^{r,t+1} = \sum_{s \in N_w} \frac{a_w^{r,t+1}}{\sum_{w' \in N_s} a_{w'}^{r,t+1}} \geq \sum_{s \in N_w} \frac{a_w^{r,t} \cdot (1+\epsilon)^{-1}}{\sum_{w' \in N_s} a_{w'}^{r,t} \cdot (1+\epsilon)} = \frac{L_w^{r,t}}{(1+\epsilon)^2}. \quad \blacktriangleleft$$

## 79:8 Distributed Load Balancing

Next, we observe that the load on any worker can never significantly exceed its target capacity.

► **Lemma 7.** *For all workers  $w$ , we have at all timesteps  $r, t$ ,*

$$L_w^{r,t} \leq (1 + 10\epsilon)(1 + \epsilon)C_w^{r,t}.$$

**Proof.** We prove the claim by induction on  $r$  and  $t$ . The base case is clearly true as

$$L_w^{0,0} \leq d = C_w^{0,0}.$$

Suppose the claim holds up to  $r, t$ . If  $0 \leq t < B$ , then Lemma 6 implies the desired for  $r, t + 1$ . If  $t = B$ , then we are in one of the following two cases

- $L_w^{r,B} < (1 + 10\epsilon)C_w^{r,B}$ , which implies  $L_w^{r+1,0} < (1 + 10\epsilon)(1 + \epsilon)C_w^{r+1,0}$
- $L_w^{r,B} \geq (1 + 10\epsilon)C_w^{r,B}$ , which implies  $C_w^{r+1,0} = C_w^{r,B}$

The first case is clearly resolved. For the second case, we can use the induction hypothesis to get the desired. ◀

When a worker becomes *fixed*, we observe that at all future timesteps, its load is close to its target capacity.

► **Lemma 8.** *For a worker  $w$ , once  $w$  is fixed, we have*

$$L_w^{r,t} \geq \frac{C_w^{r,t}}{(1 + \epsilon)^2}$$

*holds for all **future** timesteps.*

**Proof.** When  $w$  is *fixed*, we must have  $L_w > C_w$ . Combining Lemma 6 with the fact that we no longer update  $C_w$ , we get the desired. ◀

For any worker whose load is significantly lower than its capacity, we note that its weight  $a_w$  must have been increased at **every** previous timestep.

► **Lemma 9.** *Consider a worker  $w$  such that for some  $0 \leq r < A$ , if*

$$L_w^{r,B} \leq \frac{C_w^{r,B}}{(1 + \epsilon)^2},$$

*then*

$$a_w = (1 + \epsilon)^{(r+1)B},$$

*i.e. if a worker is significantly underallocated when we reach the capacity update step, then its weight must have been increased at every step.*

**Proof.** This follows immediately from Lemma 6 and the fact that the capacities  $C_w$  are weakly decreasing. ◀

## 4.2 Majorization

We present a basic inequality about convex functions that will be useful later on for bounding the objective value. We first introduce the concept of *majorization*.

► **Definition 10** (Majorization). For two sequences of real numbers  $(x_1, \dots, x_n)$  and  $(y_1, \dots, y_n)$ , let  $\pi, \sigma$  be permutations such that

$$x_{\pi(1)} \geq \dots \geq x_{\pi(n)}, y_{\sigma(1)} \geq \dots \geq y_{\sigma(n)}.$$

We say  $(x_1, \dots, x_n)$  **weakly majorizes**  $(y_1, \dots, y_n)$  if for all  $1 \leq k \leq n$

$$\sum_{i=1}^k x_{\pi(i)} \geq \sum_{i=1}^k y_{\sigma(i)}.$$

If we also have that

$$\sum_{i=1}^n x_{\pi(i)} = \sum_{i=1}^n y_{\sigma(i)}$$

then we say  $(x_1, \dots, x_n)$  **majorizes**  $(y_1, \dots, y_n)$

Intuitively, a sequence  $(x_1, \dots, x_n)$  majorizes a sequence  $(y_1, \dots, y_n)$  if the terms of  $(x_1, \dots, x_n)$  are more imbalanced. The following inequality states that a symmetric convex function takes larger values when the inputs are more imbalanced:

► **Lemma 11.** Let  $f$  be a convex function that is symmetric and non-decreasing in each of the variables. Given sequences  $(x_1, \dots, x_n)$  and  $(y_1, \dots, y_n)$  such that  $(x_1, \dots, x_n)$  weakly majorizes  $(y_1, \dots, y_n)$ , we have

$$f(x_1, \dots, x_n) \geq f(y_1, \dots, y_n).$$

If  $(x_1, \dots, x_n)$  majorizes  $(y_1, \dots, y_n)$  then the above inequality holds without the assumption that  $f$  is non-decreasing.

**Proof.** [12] proves the above inequality when  $(x_1, \dots, x_n)$  majorizes  $(y_1, \dots, y_n)$ , we adapt it to the case that  $(x_1, \dots, x_n)$  weakly majorizes  $(y_1, \dots, y_n)$ . In particular, we prove that there exists a sequence  $(y'_1, \dots, y'_n)$  such that  $y'_i \geq y_i$  for all  $1 \leq i \leq n$  and  $(x_1, \dots, x_n)$  majorizes  $(y'_1, \dots, y'_n)$ . WLOG, we assume  $x_1 \geq \dots \geq x_n$  and  $y_1 \geq \dots \geq y_n$ . We construct the sequence as follows. We first set  $y'_1$  to be the largest value so that  $(x_1, \dots, x_n)$  weakly majorizes  $(y'_1, y_2, \dots, y_n)$ . After determining the value of  $y'_1$ , we set  $y'_2$  to be the largest value so that  $(x_1, \dots, x_n)$  weakly majorizes  $(y'_1, y'_2, y_3, \dots, y_n)$ . We repeat this process to set all of  $y'_3, \dots, y'_n$ . Now, for each index  $i \in [n]$ , there exists an index  $j$  with  $j \geq i$  such that

$$x_1 + \dots + x_j = y'_1 + \dots + y'_i + y_{i+1} + \dots + y_j,$$

as otherwise, this would contradict the maximality of  $y'_i$ . Hence, we conclude that

$$x_1 + \dots + x_n = y'_1 + \dots + y'_n,$$

so  $(x_1, \dots, x_n)$  majorizes  $(y'_1, \dots, y'_n)$ , as desired. ◀

### 4.3 Main Proof

Now we are ready to analyze the performance of the algorithm. The following lemma is essential to our proof. Intuitively, it allows us to construct a set of cuts that “certify” that the solution computed by the algorithm is essentially optimal.

## 79:10 Distributed Load Balancing

► **Lemma 12.** *Let  $X_0 = \emptyset$ . We can construct sets  $X_1, \dots, X_A \subset W$  with the following properties:*

- $X_1 \subset X_2 \subset \dots \subset X_A$
- $|N(X_r)| \geq \sum_{i=0}^{r-1} |X_{i+1} \setminus X_i| \cdot \frac{d}{(1+\epsilon)^{i+10}}$
- $X_i$  contains all of the workers that are fixed after executing PROPORTIONAL ALLOCATION and capacity updates for  $r = 0, 1, \dots, i-1$

**Proof.** We prove the claim by induction. Assume that we have already constructed  $X_1, \dots, X_{i-1}$ . Let  $F^i$  be the set of workers that become *fixed* after completing the capacity updates for  $r = i-1$  but are not fixed before this. For any worker  $w \in F^i$ , we must have

$$a_w^{i-1,t+1} = \frac{a_w^{i-1,t}}{1+\epsilon} \quad \forall 0 \leq t < B.$$

This is because if this was not the case, we must have  $L_w^{i-1,t_0} < C_w^{i-1,t_0}$  for some  $t_0$  and by Lemma 6, this implies  $L_w^{i-1,t} \leq (1+\epsilon)^2 C_w^{i-1,t}$  holds for all  $t \geq t_0$ . This contradicts the fact that worker  $w$  gets fixed in the execution of the capacity updates for  $r = i-1$ . Thus, we conclude for any  $w \in F^i$

$$a_w^{i-1,B} \leq (1+\epsilon)^{(i-1)B}.$$

Let  $Y = F^i \setminus (F^i \cap X_{i-1})$ . If  $Y$  is empty then it suffices to set  $X_i = X_{i-1}$ . Assume  $Y$  is not empty, then for any integer  $0 \leq j \leq B$ , define  $Z_j$  as

$$Z_j := \{w | w \in W, a_w \leq (1+\epsilon)^{(i-1)B+j}\}.$$

We note that  $Y \subseteq Z_0 \subseteq Z_1 \subseteq \dots \subseteq Z_B$  and therefore  $Z_0 \neq \emptyset$ .

Since  $B \geq \frac{100 \log(n/\epsilon) \cdot \log(d/\epsilon)}{\epsilon^2}$ , there must exist some index  $j$  satisfying

$$10 \log\left(\frac{d}{\epsilon}\right) \cdot \frac{1}{\epsilon} \leq j < B$$

and

$$|Z_j \setminus X_{i-1}| \leq (1+\epsilon) \cdot \left| Z_{j-10 \log(\frac{d}{\epsilon}) \cdot \frac{1}{\epsilon}} \setminus X_{i-1} \right|. \quad (2)$$

We set  $X_i = Z_j \cup X_{i-1}$  and prove it satisfies all three properties in the rest of the proof. The first and third properties are clearly satisfied, i.e.,  $Z_j \cup X_{i-1}$  is a superset of  $X_{i-1}$  and it contains all of the workers that are fixed after  $i$  executions of PROPORTIONAL ALLOCATION. It remains to verify the second one

For any worker  $w \in Z_{j-10 \log(\frac{d_{\max}}{\epsilon}) \cdot \frac{1}{\epsilon}}$ , if the worker  $w$  is connected to some source  $s$ , such that  $s$  has another neighbor worker  $w'$  with  $w' \notin Z_j$ , then we have

$$x_{s,w}^{i-1,B} \leq \frac{a_w^{i-1,B}}{a_w^{i-1,B} + a_{w'}^{i-1,B}} < \frac{(1+\epsilon)^{(i-1)B+j-10 \log(\frac{d}{\epsilon}) \cdot \frac{1}{\epsilon}}}{(1+\epsilon)^{(i-1)B+j}} \leq \left(\frac{\epsilon}{d}\right)^{10}$$

Therefore, the total contributions  $x_{s,w}^{i-1,B}$  from all sources  $s$  with a neighbor not in  $Z_j$  to the worker  $w$  is at most

$$\sum_{\substack{s \in N_w \\ N_s \cap (W \setminus Z_j) \neq \emptyset}} x_{s,w}^{i-1,B} \leq \left(\frac{\epsilon}{d}\right)^{10} \cdot d \leq \left(\frac{\epsilon}{d}\right)^9. \quad (3)$$

Now we bound the changes of  $|N(X_i)|$  comparing to  $|N(X_{i-1})|$ . Consider the graph  $G^{i-1,B}$  at timestep  $(i-1, B)$ . We restrict the graph to the one induced by the set of workers  $Z_j$  and the set of sources  $N(Z_j)$ . For each worker  $w \in Z_{j-10 \log(\frac{d}{\epsilon}) \cdot \frac{1}{\epsilon}} \setminus X_{i-1}$ , its load in  $G^{i-1,B}$  satisfies

$$L_w^{i-1,B} \geq \frac{C_w^{i-1,B}}{(1+\epsilon)^2} \geq \frac{d}{(1+\epsilon)^{i+1}} \quad (4)$$

The first step comes from Lemma 9, the second step comes from the fact that  $C_w^{i-1,B} = d/(1+\epsilon)^{i-1}$ .

Thus the load of worker  $w$  in the restricted graph is at least

$$\begin{aligned} \sum_{\substack{s \in N_w \\ N_s \cap (W \setminus Z_j) = \emptyset}} x_{s,w}^{r, A_r-1, B_r} &= L_w^{r, A_r-1, B_r} - \sum_{\substack{s \in N_w \\ N_s \cap (W \setminus Z_j) \neq \emptyset}} x_{s,w}^{r, A_r-1, B_r} \\ &\geq \frac{d}{(1+\epsilon)^{i+1}} - \left(\frac{\epsilon}{d}\right)^9 \\ &\geq \frac{d}{(1+\epsilon)^{i+2}}. \end{aligned} \quad (5)$$

The second step comes from Eq. (3)(4).

Consequently, we have

$$\begin{aligned} |N(X_i)| - |N(X_{i-1})| &= |N(Z_j \cup X_{i-1})| - |N(X_{i-1})| \\ &\geq \sum_{w \in Z_j \setminus X_{i-1}} \sum_{\substack{s \in N_w \\ N_s \cap (W \setminus Z_j) = \emptyset}} x_{s,w}^{r, A_r-1, B_r} \\ &\geq \frac{d}{(1+\epsilon)^{i+2}} |Z_{j-10 \log(\frac{d}{\epsilon}) \cdot \frac{1}{\epsilon}} \setminus X_{i-1}| \\ &\geq \frac{d}{(1+\epsilon)^{i+3}} |Z_j \setminus X_{i-1}| \\ &= \frac{d}{(1+\epsilon)^{i+3}} |X_i \setminus X_{i-1}|. \end{aligned}$$

The third step follows from Eq. (5), the fourth step follows from Eq. (2) We complete the induction here.  $\blacktriangleleft$

Let  $X_0 = \emptyset, X_1, \dots, X_A$  be the sets constructed in Lemma 12. In the following two lemmas, we will compare our solution with the optimal solution using the hierarchy of certificates given by Lemma 12.

► **Lemma 13.** *Consider a feasible assignment and suppose the loads on the workers are  $L_1, \dots, L_{n_W}$ . Then the sequence  $(L_1, \dots, L_{n_W})$  weakly majorizes the following sequence  $\Gamma_1$ :*

- $|X_1 \setminus X_0|$  copies of  $\frac{d}{(1+\epsilon)^{11}}$
- $|X_2 \setminus X_1|$  copies of  $\frac{d}{(1+\epsilon)^{12}}$
- $\vdots$
- $|X_A \setminus X_{A-1}|$  copies of  $\frac{d}{(1+\epsilon)^{10+A}}$
- All remaining terms are set to 0

## 79:12 Distributed Load Balancing

**Proof.** Since the last  $n_W - |X_A|$  term of  $\Gamma_1$  is 0, it suffices to prove majorization for the first  $n_W - |X_A|$  indices. For any index  $1 \leq j \leq n_W - |X_A|$ , suppose  $|X_r| < j \leq |X_{r+1}|$  holds for some  $r$  ( $0 \leq r \leq A-1$ ). By Lemma 12, we have

$$\sum_{w \in X_{r+1}} L_w \geq \sum_{i=0}^r |X_{i+1} \setminus X_i| \cdot \frac{d}{(1+\epsilon)^{i+10}}$$

and

$$\sum_{w \in X_r} L_w \geq \sum_{i=0}^{r-1} |X_{i+1} \setminus X_i| \cdot \frac{d}{(1+\epsilon)^{i+10}}$$

Suppose  $G(r, j)$  consists of the largest  $j - |X_r|$  workers in  $X_{r+1} \setminus X_r$ , then we have

$$\begin{aligned} \sum_{w \in X_r \cup G(r, j)} L_w &\geq \sum_{w \in X_r} L_w + \frac{j - |X_r|}{|X_{r+1}| - |X_r|} \sum_{w \in X_{r+1} \setminus X_r} L_w \\ &= \frac{j - |X_r|}{|X_{r+1}| - |X_r|} \sum_{w \in X_{r+1}} L_w + \frac{|X_{r+1}| - j}{|X_{r+1}| - |X_r|} \sum_{w \in X_r} L_w \\ &\geq \frac{j - |X_r|}{|X_{r+1}| - |X_r|} \cdot \sum_{i=0}^r |X_{i+1} \setminus X_i| \cdot \frac{d}{(1+\epsilon)^{i+10}} \\ &\quad + \frac{|X_{r+1}| - j}{|X_{r+1}| - |X_r|} \sum_{i=0}^{r-1} |X_{i+1} \setminus X_i| \cdot \frac{d}{(1+\epsilon)^{i+10}} \\ &= \sum_{i=0}^{r-1} |X_{i+1} \setminus X_i| \cdot \frac{d}{(1+\epsilon)^{i+10}} + (j - |X_r|) \cdot \frac{d}{(1+\epsilon)^{r+10}} \end{aligned}$$

Thus completing the proof. ◀

► **Lemma 14.** Consider the assignment returned by our algorithm and suppose the loads on the workers are  $L'_1, L'_2, \dots, L'_{n_W}$ . Then the sequence  $(L'_1, \dots, L'_{n_W})$  is weakly majorized by the following sequence  $\Gamma_2$

- $|X_1 \setminus X_0|$  copies of  $\frac{(1+50\epsilon)d}{(1+\epsilon)^1}$
- $|X_2 \setminus X_1|$  copies of  $\frac{(1+50\epsilon)d}{(1+\epsilon)^2}$
- ⋮
- $|X_A \setminus X_{A-1}|$  copies of  $\frac{(1+50\epsilon)d}{(1+\epsilon)^A}$
- All remaining terms set to 0

**Proof.** For each index  $1 \leq i \leq A$ , define  $W^i$  to be the set of workers that are fixed after completing  $i$  full executions of PROPORTIONAL ALLOCATION and capacity update step. Consider any worker  $w$  that satisfies

$$L_w \geq \frac{(1+40\epsilon)d}{(1+\epsilon)^i}.$$

By Lemma 7, the capacity of worker  $w$  can be bounded as

$$C_w \geq \frac{(1+20\epsilon)d}{(1+\epsilon)^i}.$$

Now it is clear that this can only happen if the worker  $w$  is fixed before round  $i$ . In particular, the number of such workers is at most  $|W^i|$ . Meanwhile, we know that  $|X_i| \geq |W^i|$  by Lemma 12. Thus, if we only consider the  $|X_A|$  largest terms of the sequence  $(L'_1, \dots, L'_{n_W})$ , they are entry-wise dominated by the terms of  $\Gamma_2$ .

Since all terms after the  $|X_A|^{\text{th}}$  term of  $\Gamma_2$  are 0, to complete the proof of the weak majorization, it suffices to show that the sum of all of the terms of  $\Gamma_2$  is at least  $L'_1 + \dots + L'_{n_W}$ . Let  $U$  be the set of workers that are not *fixed* at the end of the execution of the algorithm. For all worker  $w \in U$ , due to the choice of  $A$ , we have

$$L'_w \leq (1 + 10\epsilon)C_w^{A-1, B} = \frac{(1 + 10\epsilon)d}{(1 + \epsilon)^{A-1}} \leq \frac{\epsilon}{d},$$

and therefore,

$$\sum_{w \in U} L'_w \leq \frac{\epsilon n_W}{d}. \quad (6)$$

Since the number of sources is at least  $\frac{n_W}{d}$ , we have

$$\sum_{w \in W} L'_w \geq \frac{n_W}{d}. \quad (7)$$

WLOG, we can assume  $L'_1 \geq L'_2 \geq \dots \geq L'_{n_W}$ . We then have

$$\sum_{i=0}^{A-1} |X_{i+1} \setminus X_i| \cdot \frac{(1 + 40\epsilon)d}{(1 + \epsilon)^i} \geq \sum_{j=1}^{|X_A|} L'_j \geq \sum_{w \in W \setminus U} L'_w \geq (1 - \epsilon) \sum_{w \in W} L'_w.$$

The second step holds due to the fact that the total number of workers that are fixed by the end of the algorithm is at most  $|X_A|$  (see Lemma 12). The last step follows from Eq. (6)(7).

Consequently, we have

$$\sum_{i=0}^{A-1} |X_{i+1} \setminus X_i| \cdot \frac{(1 + 50\epsilon)d}{(1 + \epsilon)^i} \geq \sum_{w \in W} L'_w,$$

which completes the proof.  $\blacktriangleleft$

Combining Lemma 13 and Lemma 14, we get

► **Corollary 15.** *Consider the assignment returned by our algorithm, and suppose the loads are  $L'_1, \dots, L'_{n_W}$ . For any feasible allocation, the loads  $L_1, \dots, L_{n_W}$  must satisfy that  $(L_1, \dots, L_{n_W})$  weakly majorizes  $\left(\frac{L'_1}{1+100\epsilon}, \dots, \frac{L'_{n_W}}{1+100\epsilon}\right)$*

Now we can easily wrap up the proof of Theorem 5.

**Proof of Theorem 5.** Let the loads in the output of our algorithm be  $L'_1, \dots, L'_{n_W}$  and let the loads in the optimal solution be  $L_1, \dots, L_{n_W}$ . Then

$$\Phi(L'_1, \dots, L'_{n_W}) \leq \Phi((1 + 100\epsilon)L_1, \dots, (1 + 100\epsilon)L_{n_W})$$

where we use Corollary 15 and Lemma 11. The above inequality immediately implies the desired.  $\blacktriangleleft$



## 5 Improved algorithm for minimizing max load

If the objective is the max function, i.e., we want to minimize the maximum load on workers, we can further reduce the number of distributed rounds to  $O\left(\frac{\log n \log d}{\epsilon^2}\right)$ . The idea is to start with a coarse estimation and gradually decrease the precision parameter down to  $\epsilon$ . The key observation is that each time we decrease the precision value, we only need to run a constant number of rounds of PROPORTIONAL ALLOCATION to refine our estimation and the number of distributed rounds of the algorithm is dominated by the number of rounds for the smallest precision parameter, which is  $O\left(\frac{\log n \log d}{\epsilon^2}\right)$ .

In Appendix B, we present an example showing that our analysis of our algorithm is tight. More specifically, we show that any proportional allocation based algorithm that starts from a uniform initialization and has step size bounded by  $\epsilon$  must run at least  $\Omega\left(\frac{\log n \log d}{\epsilon^2}\right)$  rounds to compute a  $1 + \epsilon$ -approximation.

To facilitate the presentation, we first define the precision sequence we use in the algorithm.

► **Definition 16.** We define the sequence  $\{\epsilon_r\}_{r \in \mathbb{N}}$  and  $\{B_r\}_{r \in \mathbb{N}}$  as follow.

■ For any  $r \geq 0$ ,  $\epsilon_{r+1} = (1 + \epsilon_r)^{1/2} - 1$  and  $\epsilon_0 = d - 1$ .

■ For any  $r \geq 0$ ,  $B_r = 2 \log_{1+\epsilon_r}(n) \log_{1+\epsilon_r}(d/\epsilon) + 8 \log_{1+\epsilon_r}(n)$

We further define  $R(n, \epsilon)$  to be the smallest index such that  $(1 + \epsilon_r)^5 \leq 1 + \epsilon$ .

### Algorithm 3 LOAD BALANCING FOR THE MAX OBJECTIVE.

---

```

1: Initialize weights  $a_w = 1$  for all workers  $w \in W$ .
2: Initialize capacity upper bounds  $C = d_{\max}$ .
3: Set  $R = R(n, \epsilon)$ .
4: for  $r = 0, 1, \dots, R - 1$  do
5:   Reset  $a_w = 1, \forall w \in W$ 
6:   while True do
7:     Run PROPORTIONAL ALLOCATION for  $B_r$  rounds with initial parameters  $a_w, \epsilon_r$ ,
        $C_w = C \forall w$ ,
8:     if  $\exists w \in W$  such that  $L_w \geq (1 + \epsilon_r)^4 C$  then
9:       Break the while loop
10:    else
11:       $C \leftarrow \frac{C}{1 + \epsilon_r}$ 
12:     $C \leftarrow C(1 + \epsilon_r)^8$ 
13: for  $s, w$  do
14:   Output  $x_{s,w} = \frac{a_w}{\sum_{w \in N_s} a_w}$ 

```

---

► **Theorem 17.** When the objective is the max function, Algorithm 3 returns an  $(1 + \epsilon)$  approximate solution to the load balancing problem after  $O\left(\frac{\log(n) \log(d/\epsilon)}{\epsilon^2}\right)$  iterations.

### Notations

For any  $0 \leq r \leq R - 1$ , we use  $A_r$  to denote the number of calls we make to PROPORTIONAL ALLOCATION within the while loop of round  $r$ . For the load variable  $L_w$ , we slightly abuse of notation and for any  $0 \leq r \leq R - 1, 0 \leq a \leq A_r - 1, 0 \leq t \leq B_r$ , we use  $L_w^{r,a,t}$  to denote

the load on worker  $w$  when the algorithm is executed to the timestep  $r, a, t$ , i.e. we finished  $r$  outer loops and completed  $a$  full iterations of PROPORTIONAL ALLOCATION, and then  $t$  rounds within the next iteration of PROPORTIONAL ALLOCATION. We apply the same notation for  $a_w$  and  $x_{s,w}$ . For the capacity variable, since we maintain the same capacity over all workers and the capacity does not change during the execution of PROPORTIONAL ALLOCATION, we simplify the notation and use  $C^{r,a}$  to denote the capacity on the worker side right after we finished  $r$  outer loops and completed  $a$  full iterations of PROPORTIONAL ALLOCATION *but before we perform the update on the capacity*.

The following observation, a restatement of Lemma 6, still holds.

- **Lemma 18.** *For any  $0 \leq r \leq R-1, 0 \leq a \leq A_r-1, 0 \leq t \leq B_r-1$*
- *If  $L_w^{r,a,t} > C^{r,a}$  then  $\frac{L_w^{r,a,t}}{(1+\epsilon_r)^2} \leq L_w^{r,a,t+1} \leq L_w^{r,a,t}$*
  - *If  $L_w^{r,a,t} < C^{r,a}$  then  $L_w^{r,a,t} \leq L_w^{r,a,t+1} \leq (1+\epsilon_r)^2 L_w^{r,a,t}$*

The following Lemma follows immediately from Lemma 18.

- **Lemma 19.** *Consider a worker  $w$  such that for some  $0 \leq r < R, 0 \leq a < A_r$ ,*

$$L_w^{r,a,B_r} \leq \frac{C_w^{r,a}}{(1+\epsilon_r)^2}$$

*then*

$$a_w = (1+\epsilon_r)^{(a+1)B_r}$$

*i.e. if a worker is significantly underallocated then its weight must have been increased at every step.*

We proceed next to the following key lemma, which says when we break the while loop for any round  $r$ , we actually find a certificate lower bound on the optimal value. The proof of Lemma 20 bares some similarities with Lemma 12, the difference is that we need to deal with the case that  $\epsilon_r$  is large. The detailed proof is provided in Appendix A for completeness.

- **Lemma 20.** *For any  $0 \leq r < R$ , we can find a subset of worker  $W_r \subseteq W$  such that*

$$|N(W_r)| \geq |W_r| \cdot \frac{C^{r,A_r}}{(1+\epsilon_r)^4}.$$

We next show that at the end of each round of PROPORTIONAL ALLOCATION, the load on any worker can never significantly exceed its target capacity.

- **Lemma 21.** *For all workers  $w$ , we have at all timesteps  $r, a$  with  $0 \leq r < R, 0 \leq a \leq A_r-1$*

$$L_w^{r,a,B_r} \leq (1+\epsilon_r)^5 C^{r,a+1}.$$

**Proof.** We prove by induction on  $(r, a)$ . The base case hold trivially as  $L_w^{0,0,B_r} \leq d_{\max} = C^{0,1}(1+\epsilon_0)$ . Suppose the claim holds up to  $(r, a)$ , it is easy to see it holds for  $(r, a+1)$ . This is due to Lemma 18 and the fact that we stop if  $L_w \geq (1+\epsilon_r)^4 C$  for any worker.

It remains to show that if the claim holds up to  $(r, A_r-1)$ , then it also holds for  $(r+1, 0)$ . This would complete the proof. We prove by contradiction. Suppose  $L_w^{r+1,0,B_r} \geq (1+\epsilon_r)^5 C^{r+1,1}$ , then we know that we need to break out the loop and followed by Lemma 20, there exists a subset of workers  $W_{r+1} \subseteq W$  such that

$$|N(W_{r+1})| \geq |W_{r+1}| \cdot \frac{C^{r+1,1}}{(1+\epsilon_{r+1})^4}$$

This actually says that the optimal solution is at least

$$\text{OPT} \geq \frac{C^{r+1,1}}{(1+\epsilon_{r+1})^4} = \frac{C^{r+1,1}}{(1+\epsilon_r)^2} = C^{r,A_r}(1+\epsilon_r)^6.$$

However, by induction, we have that  $L_w^{r,A_r-1,B_r} \leq (1+\epsilon_r)^5 C^{r,A_r}$ . this says that we actually have a solution with max load bounded by  $(1+\epsilon_r)^5 C^{r,A_r}$ , i.e.,  $\text{OPT} \leq (1+\epsilon_r)^5 C^{r,A_r}$ . This comes to a contradiction and we conclude the proof here.  $\blacktriangleleft$

► **Lemma 22.** For  $0 \leq r < R$ , the optimal solution satisfies

$$\text{OPT} \in [(1+\epsilon_r)^{-4} C^{r,A_r}, (1+\epsilon_r)^5 C^{r,A_r}]$$

**Proof.** The lower bound comes from Lemma 20. The upper bound comes from Lemma 21.  $\blacktriangleleft$

We can now wrap up the proof of Theorem 17

**Proof of Theorem 17.** The correctness of the algorithm comes as a direct corollary of Lemma 22. For the running time, for any  $0 \leq r < R$ , we claim that  $A_r \leq 30$ . The claim holds trivially for  $r = 0$  and for  $r \geq 1$ , we have

$$\begin{aligned} \text{OPT} &< (1+\epsilon_r)^6 C^{r,A_r} = (1+\epsilon_r)^{6-A_r} C^{r,0} = (1+\epsilon_{r-1})^{3-A_r/2} C^{r,0} \\ &= (1+\epsilon_{r-1})^{11-A_r/2} C^{r-1,A_{r-1}} \leq (1+\epsilon_{r-1})^{15-A_r/2} \text{OPT} \end{aligned}$$

The first inequality follows from the upper bound of Lemma 21, the second step follows from  $C^{r,A_r} = (1+\epsilon)^{-A_r} C^{r,0}$ . The third step follows from the fact that  $(1+\epsilon_r) = (1+\epsilon_{r-1})^{1/2}$ , the fourth step follows from the fact that  $C^{r,0} = (1+\epsilon_{r-1})^8 C^{r-1,A_{r-1}}$ , the last inequality follows from the lower bound of Lemma 21.

The total number of iterations is then bounded by

$$\begin{aligned} \sum_{r=0}^R A_r B_r &\leq 30 \sum_{r=0}^R B_r \lesssim \sum_{r=0}^R \log_{1+\epsilon_r}(n) \log_{1+\epsilon_r}(d/\epsilon_r) + \log_{1+\epsilon_r}(n) \\ &= \sum_{r=0}^{\epsilon_r \geq 1} \log_{1+\epsilon_r}(n) \log_{1+\epsilon_r}(d/\epsilon_r) + \sum_{r:\epsilon_r \leq 1}^R \log_{1+\epsilon_r}(n) \log_{1+\epsilon_r}(d/\epsilon_r) \\ &\lesssim \sum_{r=0}^{\epsilon_r \geq 1} \log_{1+\epsilon_r}(n) \log_{1+\epsilon_r}(d/\epsilon_r) + \sum_{r:\epsilon_r \leq 1}^R \frac{\log n \log(d/\epsilon)}{\epsilon_r^2} \\ &\lesssim \log n \log d + \frac{\log n \log(d/\epsilon)}{\epsilon^2} \\ &= O\left(\frac{\log n \log(d/\epsilon)}{\epsilon^2}\right) \end{aligned}$$

Thus completing the proof.  $\blacktriangleleft$

---

## References

- 1 Shipra Agrawal, Vahab Mirrokni, and Morteza Zadimoghaddam. Proportional allocation: Simple, distributed, and diverse matching with high entropy. In *International Conference on Machine Learning*, pages 99–108, 2018.
- 2 Zeyuan Allen-Zhu and Lorenzo Orecchia. Using optimization to break the epsilon barrier: A faster and simpler width-independent algorithm for solving positive linear programs in parallel. In *Proceedings of the twenty-sixth annual ACM-SIAM symposium on Discrete algorithms*, pages 1439–1456. SIAM, 2014.

- 3 Zeyuan Allen-Zhu and Lorenzo Orecchia. Nearly linear-time packing and covering lp solvers. *Mathematical Programming*, 175(1-2):307–353, 2019.
- 4 Sepehr Assadi, Aaron Bernstein, and Zachary Langley. Improved bounds for distributed load balancing. *arXiv preprint*, 2020. [arXiv:2008.04148](https://arxiv.org/abs/2008.04148).
- 5 Baruch Awerbuch and Rohit Khandekar. Stateless distributed gradient descent for positive linear programs. *SIAM Journal on Computing*, 38(6):2468–2486, 2009.
- 6 Petra Berenbrink, Tom Friedetzky, and Zengjian Hu. A new analytical method for parallel, diffusion-type load balancing. *Journal of Parallel and Distributed Computing*, 69(1):54–61, 2009.
- 7 Petra Berenbrink, Tom Friedetzky, and Russell Martin. Dynamic diffusion load balancing. In *International Colloquium on Automata, Languages, and Programming*, pages 1386–1398. Springer, 2005.
- 8 Andrzej Czygrinow, Michal Hanćkowiak, Edyta Szymańska, and Wojciech Wawrzyniak. Distributed 2-approximation algorithm for the semi-matching problem. In *International Symposium on Distributed Computing*, pages 210–222. Springer, 2012.
- 9 Magnús M Halldórsson, Sven Köhler, Boaz Patt-Shamir, and Dror Rawitz. Distributed backup placement in networks. *Distributed Computing*, 31(2):83–98, 2018.
- 10 Michael Luby and Noam Nisan. A parallel approximation algorithm for positive linear programming. In *Proceedings of the twenty-fifth annual ACM symposium on Theory of computing*, pages 448–457, 1993.
- 11 Michael W Mahoney, Satish Rao, Di Wang, and Peng Zhang. Approximating the solution to mixed packing and covering lps in parallel  $o(\epsilon^{-3})$  time. In *43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2016.
- 12 Albert W Marshall and Frank Proschan. An inequality for convex functions involving majorization. Technical report, BOEING SCIENTIFIC RESEARCH LABS SEATTLE WASH, 1964.
- 13 Yuval Rabani, Alistair Sinclair, and Rolf Wanka. Local divergence of markov chains and the analysis of iterative load-balancing schemes. In *Proceedings 39th Annual Symposium on Foundations of Computer Science (Cat. No. 98CB36280)*, pages 694–703. IEEE, 1998.
- 14 Google Search Statistics. *Internet Live Stats*, 2020. URL: <https://www.internetlivestats.com/google-search-statistics/>.
- 15 Raghu Subramanian and Isaac D Scherson. An analysis of diffusive load-balancing. In *Proceedings of the sixth annual ACM symposium on Parallel algorithms and architectures*, pages 220–225, 1994.
- 16 Di Wang. *Fast Approximation Algorithms for Positive Linear Programs*. PhD thesis, UC Berkeley, 2017.
- 17 Di Wang, Satish Rao, and Michael W Mahoney. Unified acceleration method for packing and covering problems via diameter reduction. In *43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2016.
- 18 Neal E Young. Sequential and parallel algorithms for mixed packing and covering. In *Proceedings 42nd IEEE symposium on foundations of computer science*, pages 538–546. IEEE, 2001.

## A

 Missing proof from Section 5

**Proof of Lemma 20.** We prove the lemma for a fixed  $r$ . Define

$$F := \{w | w \in W, L_w^{r, A_r-1, B_r} > (1 + \epsilon_r)^4 C^{r, A_r}\},$$

i.e.,  $F$  is the subset of workers whose load significantly outweigh the target capacity and break the while loop. For any worker  $w \in F$ , we assert that its weight  $a_w$  must decrease every timestep during the last call to PROPORTIONAL ALLOCATION, i.e.,

$$a_w^{r, A_r-1, t+1} = \frac{a_w^{r, A_r-1, t}}{1 + \epsilon_r} \quad \forall 0 \leq t < B_r.$$

This is because if  $L_w^{r, A_r-1, t_0} < C^{r, A_r-1}$  for some  $t_0$ , then by Lemma 18, we know that  $L_w^{r, A_r-1, t} \leq (1 + \epsilon_r)^2 C^{r, A_r-1}$  holds for all  $t \geq t_0$ , which contradicts with the fact that  $L_w^{r, A_r-1, B_r} > (1 + \epsilon_r)^4 C^{r, A_r}$ . Thus we conclude that

$$a_w^{A_r-1, B_r} \leq (1 + \epsilon_r)^{(A_r-1)B_r} \quad \forall w \in F.$$

For any  $0 \leq j \leq B_r$ , define

$$Z_j = \left\{ w | w \in W, a_w^{A_r-1, B_r} \leq (1 + \epsilon_r)^{(A_r-1)B_r+j} \right\}.$$

We note that  $F \subseteq Z_0 \subseteq Z_1 \cdots \subseteq Z_{B_r} = W$  and therefore  $Z_0 \neq \emptyset$ .

Since  $B_r = 2 \log_{1+\epsilon_r}(n) \log_{1+\epsilon_r}(d/\epsilon) + 8 \log_{1+\epsilon_r}(n)$ , there must exist some index  $j$  satisfies

$$2 \log_{1+\epsilon_r}(d/\epsilon) + 8 \leq j < B_r$$

and

$$|Z_j| \leq (1 + \epsilon_r)^{|Z_{j-2 \log_{1+\epsilon_r}(d/\epsilon)-8}|}. \quad (8)$$

We set  $W_r = Z_j$  and will show that  $|N(Z_j)| \geq |Z_j| \cdot C^{r, A_r} / (1 + \epsilon_r)^4$  in the rest of the proof. For any worker  $w \in Z_{j-2 \log_{1+\epsilon_r}(d/\epsilon)-8}$ , if worker  $w$  is connected to some source  $s$  such that  $s$  has another neighbor  $w'$  with  $w' \notin Z_j$ . Then we have

$$\begin{aligned} x_{s,w}^{r, A_r-1, B_r} &\leq \frac{a_w^{r, A_r-1, B_r}}{a_w^{r, A_r-1, B_r} + a_{w'}^{r, A_r-1, B_r}} \leq \frac{(1 + \epsilon_r)^{(A_r-1)B_r+j-2 \log_{1+\epsilon_r}(d/\epsilon)-8}}{(1 + \epsilon_r)^{(A_r-1)B_r+j+1}} \\ &\leq \frac{\epsilon_r^2}{d^2(1 + \epsilon_r)^9} \leq \frac{\epsilon_r}{d^2(1 + \epsilon_r)^8}. \end{aligned}$$

Hence, the total contributions  $x_{s,w}^{r, A_r-1, B_r}$  from all sources  $s$  with a neighbor not in  $Z_j$  to worker  $w$  is at most

$$\sum_{\substack{s \in N_w \\ N_s \cap (W \setminus Z_j) \neq \emptyset}} x_{s,w}^{r, A_r-1, B_r} \leq \frac{\epsilon_r}{d^2(1 + \epsilon_r)^8} \cdot d \leq \frac{\epsilon_r}{d(1 + \epsilon_r)^8}. \quad (9)$$

Consider the restricted graph between the source set  $N(Z_j)$  and the worker set  $Z_j$ , for each worker  $w \in Z_{j-2 \log_{1+\epsilon_r}(d/\epsilon)-8}$ , by Lemma 19, we know that its load satisfies

$$L_w^{r, A_r-1, B_r} \geq \frac{C^{r, A_r-1}}{(1 + \epsilon_r)^2}.$$

Hence the load of worker  $w$  in the restricted graph is at least

$$\begin{aligned}
\sum_{\substack{s \in N_w \\ N_s \cap (W \setminus Z_j) = \emptyset}} x_{s,w}^{r, A_r-1, B_r} &= L_w^{r, A_r-1, B_r} - \sum_{\substack{s \in N_w \\ N_s \cap (W \setminus Z_j) \neq \emptyset}} x_{s,w}^{r, A_r-1, B_r} \\
&\geq \frac{C^{r, A_r}}{(1 + \epsilon_r)^2} - \frac{\epsilon_r}{d_{\max}(1 + \epsilon_r)^8} \\
&\geq \frac{C^{r, A_r}}{(1 + \epsilon_r)^3}
\end{aligned} \tag{10}$$

The second step follows from Eq. (9), the last step follows from  $C^{r, A_r} \geq (1 + \epsilon_r)^{-5}$  holds all the time.

Thus, we have

$$\begin{aligned}
|N(Z_j)| &= \sum_{w \in Z_j} \sum_{\substack{s \in N_w \\ N_s \cap (W \setminus Z_j) = \emptyset}} x_{s,w}^{r, A_r-1, B_r} \\
&\geq \sum_{w \in Z_j - 2 \log_{1+\epsilon_r}(d_{\max}/\epsilon) - 4} \sum_{\substack{s \in N_w \\ N_s \cap (W \setminus Z_j) = \emptyset}} x_{s,w}^{r, A_r-1, B_r} \\
&\geq |Z_j - 2 \log_{1+\epsilon_r}(d_{\max}/\epsilon) - 4| \cdot \frac{C^{r, A_r}}{(1 + \epsilon_r)^3} \\
&\geq \frac{C^{r, A_r}}{(1 + \epsilon_r)^4} \cdot |Z_j|.
\end{aligned}$$

The third step follows from Eq. (10), the last step follows from Eq. (9). We complete the proof here.  $\blacktriangleleft$

## B Tightness

Here we present an example showing that our analysis of our algorithm is tight for the case of the max function even if we know the optimum value in advance. More formally, we show that starting from a state where all of the weights are initialized to 1 and the precision parameter is set to  $\epsilon$ , running  $\Omega\left(\frac{\log^2 n}{\epsilon^2}\right)$  iterations of proportional allocation is actually necessary.

Note here we assume  $1/\epsilon = n^{o(1)}$  i.e. the desired tolerance is much larger than  $n^{-1}$ .

Consider the following graph  $G$ . Let  $k = 0.5 \log n / \epsilon$ . We have sets  $S_1, S_2, \dots, S_k$  of sources and sets  $W_1, W_2, \dots, W_k$  of workers. For all  $i$  let

$$|S_i| = |W_i| = (1 - \epsilon)^i \epsilon n.$$

In  $G$ , there is a perfect matching between the vertices of  $W_i$  and the vertices of  $S_i$  for all  $1 \leq i \leq k$ . Also, there is a complete bipartite graph between the vertices of  $S_i$  and  $W_{i+1}$  for  $1 \leq i \leq k - 1$ .

Consider the initial state where all of the sources distribute their load uniformly among the adjacent workers. In this case the maximum load among all of the workers is

$$1 + \frac{|S_{k-1}| \cdot |W_k|}{|W_k| + 1} \geq 2.$$

## 79:20 Distributed Load Balancing

However, in the optimal allocation, each set of sources  $S_i$  assigns all of its load to the workers in  $W_i$  according to the perfect matching and the maximum load among all of the workers is 1. Now we will show that for some positive constant  $c$ , starting from the initial state where all sources distribute their load uniformly, after  $\frac{c \log^2 n}{\epsilon^2}$  iterations of proportional allocation, the maximum load among all of the workers is at least  $1 + 0.1\epsilon$ .

First observe that within each set  $W_i$ , by symmetry, all of the weights of the workers are equal at all times. Let  $w_i^t$  denote the weight of a worker in  $W_i$  after  $t$  rounds. If there exists an  $i$  such that

$$w_i^t \geq w_{i-1}^t n^{-0.1}$$

then consider the sources in  $S_{i-1}$ . The amount of their load going to the workers in  $W_i$  is at least  $\frac{|S_{i-1}| \cdot |W_i|}{|W_i| + n^{0.1}} \geq |S_i|$ . Thus the total load going to workers in  $W_k \cup W_{k-1} \cup \dots \cup W_i$  is at least

$$|S_k| + \dots + |S_i| + |S_i|.$$

Note that  $|S_k|, |S_{k-1}|, \dots, |S_i|$  is a geometric series with ratio  $\frac{1}{1-\epsilon}$  so

$$|S_i| \geq 0.1\epsilon(|S_k| + \dots + |S_i|).$$

Thus, there is some worker among  $W_k \cup W_{k-1} \cup \dots \cup W_i$  with load at least

$$\frac{(1 + 0.1\epsilon)(|S_k| + \dots + |S_i|)}{(|W_k| + \dots + |W_i|)} = 1 + 0.1\epsilon.$$

On the other hand, if we have

$$w_i^t \leq w_{i-1}^t n^{-0.1}$$

for all  $i$ , then the ratio between the smallest and largest weight among all of the workers must be at least  $n^{0.1k}$ . However, this ratio is 1 at the beginning and increases by a factor of at most  $(1 + \epsilon)^2$  each round so the number of rounds must be at least

$$\Omega\left(\frac{\log n}{\epsilon} k\right) = \Omega\left(\frac{\log^2 n}{\epsilon^2}\right).$$

► **Remark 23.** Based on this example, an obvious modification would be to adaptively increase the step size of the proportional allocation updates for certain workers depending on their current load. However, note that initially all workers that are not in  $W_1$  or  $W_k$  have load very close to 1. In other words, for every worker, there is another worker in its two-hop neighborhood whose load is very close to the optimal load of 1. If we significantly increased the step size, then we would no longer have the stability conditions (Lemma 7 and Lemma 21) and our analysis would have to be quite different.



# Erasure-Resilient Sublinear-Time Graph Algorithms

**Amit Levi**

David R. Cheriton School of Computer Science, University of Waterloo, Canada  
amit.levi@uwaterloo.ca

**Ramesh Krishnan S. Pallavoor** 

Department of Computer Science, Boston University, MA, USA  
rameshkp@bu.edu

**Sofya Raskhodnikova**

Department of Computer Science, Boston University, MA, USA  
sofya@bu.edu

**Nithin Varma** 

Department of Computer Science, University of Haifa, Israel  
nvarma@bu.edu

---

## Abstract

We investigate sublinear-time algorithms that take partially erased graphs represented by adjacency lists as input. Our algorithms make degree and neighbor queries to the input graph and work with a specified fraction of adversarial erasures in adjacency entries. We focus on two computational tasks: testing if a graph is connected or  $\varepsilon$ -far from connected and estimating the average degree. For testing connectedness, we discover a threshold phenomenon: when the fraction of erasures is less than  $\varepsilon$ , this property can be tested efficiently (in time independent of the size of the graph); when the fraction of erasures is at least  $\varepsilon$ , then a number of queries linear in the size of the graph representation is required. Our erasure-resilient algorithm (for the special case with no erasures) is an improvement over the previously known algorithm for connectedness in the standard property testing model and has optimal dependence on the proximity parameter  $\varepsilon$ . For estimating the average degree, our results provide an “interpolation” between the query complexity for this computational task in the model with no erasures in two different settings: with only degree queries, investigated by Feige (SIAM J. Comput. ‘06), and with degree queries and neighbor queries, investigated by Goldreich and Ron (Random Struct. Algorithms ‘08) and Eden et al. (ICALP ‘17). We conclude with a discussion of our model and open questions raised by our work.

**2012 ACM Subject Classification** Theory of computation → Streaming, sublinear and near linear time algorithms; Mathematics of computing → Approximation algorithms

**Keywords and phrases** Graph property testing, Computing with incomplete information, Approximating graph parameters

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.80

**Related Version** The full version of this article is available at <https://arxiv.org/abs/2011.14291> [20].

**Funding** *Amit Levi*: Part of this work was done while the author was visiting Boston University.  
*Ramesh Krishnan S. Pallavoor*: The work of this author was partially supported by NSF award CCF-1909612 and was done in part while the author was visiting the Simons Institute for the Theory of Computing.  
*Sofya Raskhodnikova*: The work of this author was partially supported by NSF award CCF-1909612 and was done in part while the author was visiting the Simons Institute for the Theory of Computing.  
*Nithin Varma*: The work of this author was partially supported by ISF grant 497/17 and Israel PBC Fellowship for Outstanding Postdoctoral Researchers from India and China. This work was done in part while the author was a student at Boston University.

**Acknowledgements** We thank Talya Eden for useful discussions that led to simplification of analysis in Section 3.1.



© Amit Levi, Ramesh Krishnan S. Pallavoor, Sofya Raskhodnikova, and Nithin Varma;  
licensed under Creative Commons License CC-BY  
12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 80; pp. 80:1–80:20



Leibniz International Proceedings in Informatics  
LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

## 1 Introduction

The goal of this work is to model and investigate sublinear-time algorithms that run on graphs with incomplete information. Typically, sublinear-time models assume that algorithms have query or sample access to an input graph. However, this assumption does not accurately reflect reality in some situations. Consider, for example, the case of a social network where vertices represent individuals and edges represent friendships. Individuals might want to hide their friendship relations for privacy reasons. When input graphs are represented by their adjacency lists, such missing information can be modeled as *erased* entries in the lists. In this work, we initiate an investigation of sublinear-time algorithms whose inputs are graphs represented by the adjacency lists with some of the entries adversarially erased.

In our erasure-resilient model of sublinear-time graph algorithms, an algorithm gets a parameter  $\alpha \in [0, 1]$  and query access to the adjacency lists of a graph with at most an  $\alpha$  fraction of the entries in the adjacency lists erased. We call such a graph  $\alpha$ -*erased* or, when  $\alpha$  is clear from the context, *partially erased*. Algorithms access partially erased graphs via degree and neighbor queries. The answer to a degree query  $v$  is the degree of the vertex  $v$ . A neighbor query is of the form  $(v, i)$ , and the answer is the  $i^{\text{th}}$  entry in the adjacency list of  $v$ . If the  $i^{\text{th}}$  entry is erased<sup>1</sup>, the answer is a special symbol  $\perp$ . A *completion* of a partially erased graph  $G$  is a valid graph represented by adjacency lists (with no erasures) that coincide with the adjacency lists of  $G$  on all nonerased entries. We formulate our computational tasks in terms of valid completions of partially erased input graphs and analyze the performance of our erasure-resilient algorithms in the *worst case* over all  $\alpha$ -erased graphs. We investigate representative problems from two fundamental classes of computational tasks in our model: graph property testing and estimating a graph parameter.

In the context of graph property testing [15], we study the problem of testing whether a partially erased graph is connected. Our model is a generalization of the *general graph model* of Parnas and Ron [23] (which is in turn a generalization of the *bounded degree model* of Goldreich and Ron [16]) to the setting with erasures. A partially erased graph  $G$  has property  $\mathcal{P}$  (in our case, is connected) if there exists a completion of  $G$  that has the property. For  $\varepsilon \in (0, 1)$ , such a graph with  $m$  edges (more precisely,  $2m$  entries in its adjacency lists) is  $\varepsilon$ -far from  $\mathcal{P}$  (in our case, from being connected) if every completion of  $G$  is different in at least  $\varepsilon m$  edges from every graph with the property. The goal of a testing algorithm is to distinguish, with high probability,  $\alpha$ -erased graphs that have the property from those that are  $\varepsilon$ -far. For testing connectedness in our erasure-resilient model, we discover a threshold phenomenon: when the fraction of erasures is less than  $\varepsilon$ , this property can be tested efficiently (in time independent of the size of the graph); when the fraction of erasures is at least  $\varepsilon$ , then a number of queries linear in the size of the graph is required to test connectedness. Additionally, when there are no erasures, our tester has better query complexity than the best previously known standard tester for connectedness [23, 5], also mentioned in the book on property testing by Goldreich [14]. Our tester has optimal dependence on  $\varepsilon$ , as evidenced by a recent lower bound in [21] for this fundamental property.

Next, we study erasure-resilient algorithms for estimating the average degree of a graph. The problem of estimating the average degree of a graph, in the case with no erasures, was studied by Feige [13], Goldreich and Ron [17], and Eden et al. [9, 10]. Feige designed an algorithm that, for all  $\varepsilon > 0$ , makes  $O(\sqrt{n}/\varepsilon)$  degree queries to an  $n$ -node graph and outputs, with high probability, an estimate that is within a factor of  $2 + \varepsilon$  of the average

<sup>1</sup> One can consider a more general model where the degrees of some vertices can also be erased. Our algorithms continue to work in this model, since one can determine the degree of a vertex using  $O(\log n)$  neighbor queries (irrespective of whether these queries are made to erased adjacency entries).

degree. He also showed that to get a 2-approximation, one needs  $\Omega(n)$  degree queries. Goldreich and Ron proved that if an algorithm can make uniformly random neighbor queries (that is, obtain a uniformly random neighbor of a specified vertex) then, for all  $\varepsilon > 0$ , the average degree can be estimated to within a factor of  $1 + \varepsilon$  using  $O(\sqrt{n} \cdot \text{poly}(\log n, 1/\varepsilon))$  queries. Eden et al. proved a tighter bound of  $O(\sqrt{n} \cdot \log \log n \cdot \text{poly}(1/\varepsilon))$  on the query complexity of this problem and provided a simpler analysis. We describe an algorithm that estimates the average degree of  $\alpha$ -erased graphs to within a factor of  $1 + \min(2\alpha, 1) + \varepsilon$  using  $O(\sqrt{n} \cdot \log \log n \cdot \text{poly}(1/\varepsilon))$  queries. Our result can be thought of as an interpolation between the results in [13] and [17, 9, 10]. In particular, when there are no erasures, that is, when  $\alpha = 0$ , we get a  $(1 + \varepsilon)$ -approximation; when all adjacency entries are erased, and only the degree queries are useful, that is, when  $\alpha = 1$ , we obtain a  $(2 + \varepsilon)$ -approximation. We also show that our result cannot be improved significantly: to get a  $(1 + \alpha)$ -approximation,  $\Omega(n)$  queries are necessary.

## Discussion of our model

For the case of graph property testing, our model is an adaptation of the erasure-resilient model for testing properties of functions by Dixit et al. [7]. Dixit et al. designed erasure-resilient testers for many properties of functions, including monotonicity, the Lipschitz property, and convexity. The conceptual difference between the two models is that the adjacency lists representation of a graph cannot be viewed as a function. (This is not the case for the adjacency matrix representation.) For a function, erased entries can be filled in arbitrarily and, as a result, they never contribute to the distance to the property. For the adjacency lists representation, this is not the case: erasures have to be filled so that the resulting completion is a valid graph. The restrictions on how they can be filled may result in some contribution to the distance coming from the erased entries<sup>2</sup>. For example, consider the property of bipartiteness. Let  $B$  be a complete balanced bipartite graph  $(U, V; E)$ , and let  $B'$  be obtained from  $B$  by adding an erased entry to the adjacency list of every vertex in  $U$ . Then, in every completion of  $B'$ , all formerly erased entries have to be changed to make the graph bipartite.

Furthermore, Dixit et al. [7] gave results only on property testing in the erasure-resilient model. We go beyond property testing in our exploration of erasure-resilient algorithms by considering more general computational tasks.

Finally, our model opens up many new research directions, some of which are discussed in Section 4.

## 1.1 The Model

We consider simple undirected graphs  $G = (V, E)$  represented by adjacency lists, where some entries in the adjacency lists could be adversarially erased (these entries are denoted by  $\perp$ ).

► **Definition 1.1** ( $\alpha$ -erased graph; completion). *Let  $\alpha \in [0, 1]$  be a parameter. An  $\alpha$ -erased graph on a vertex set  $V$  is a concatenation of the adjacency lists of a simple undirected graph  $(V, E)$  with at most an  $\alpha$  fraction of all entries (that is, at most  $2\alpha|E|$  entries) in the lists erased. A completion of an  $\alpha$ -erased graph  $G$  is the adjacency lists representation of a simple undirected graph  $G'$  that coincides with  $G$  on all nonerased entries.*

<sup>2</sup> Because of this, we make an adjustment to the model of Dixit et al. [7]: we measure the distance to the property as a fraction of the completion representation that needs to be changed, as opposed to the fraction of the nonerased representation that needs to be changed.

By definition, every partially erased graph has a completion, because it was obtained by erasing entries in a valid graph.

Given a partially erased graph  $G$  over a vertex set  $V$ , we use  $n$  to denote  $|V|$  and  $m$  to denote the number of edges in any completion of  $G$ , that is, half the sum of lengths of the adjacency lists of all the vertices in  $G$ . The average degree, that is,  $2m/n$ , is denoted by  $\bar{d}$ . For  $u \in V$ , we use  $\text{Adj}(u)$  to denote the adjacency list of  $u$ . The degree  $u$ , denoted  $\deg(u)$ , is the length of  $\text{Adj}(u)$ .

► **Definition 1.2** (Nonerased and half-erased edges). *Let  $G$  be a partially erased graph over a vertex set  $V$ . For vertices  $u, v \in V$ , the set  $\{u, v\}$  is a nonerased edge in  $G$  if  $u$  is present in  $\text{Adj}(v)$  and vice versa. The set  $\{u, v\}$  is a half-erased edge if  $u$  is in  $\text{Adj}(v)$  but  $v$  is not in  $\text{Adj}(u)$ , or vice versa.*

Our algorithms make two types of queries: *degree queries* and *neighbor queries*. A degree query specifies a vertex  $v$ , and the answer is  $\deg(v)$ . A neighbor query specifies  $(v, i)$ , and the answer is the  $i^{\text{th}}$  entry in  $\text{Adj}(v)$ .

► **Definition 1.3** (Distance to a property; erasure-resilient property tester). *Let  $\alpha \in [0, 1]$ ,  $\varepsilon \in (0, 1)$  be parameters. An  $\alpha$ -erased graph  $G$  satisfies a property  $\mathcal{P}$  if there exists a completion of  $G$  that satisfies  $\mathcal{P}$ . An  $\alpha$ -erased graph  $G$  is  $\varepsilon$ -far from a property  $\mathcal{P}$  if every completion  $G'$  of  $G$  is different in at least  $\varepsilon m$  edges from every graph that satisfies  $\mathcal{P}$ .*

*An  $\alpha$ -erasure-resilient  $\varepsilon$ -tester for a property  $\mathcal{P}$  gets parameters  $\alpha \in [0, 1]$ ,  $\varepsilon \in (0, 1)$  and query access to an  $\alpha$ -erased graph  $G$ . The tester accepts, with probability at least  $2/3$ , if  $G$  satisfies  $\mathcal{P}$ . The tester rejects, with probability at least  $2/3$ , if  $G$  is  $\varepsilon$ -far from  $\mathcal{P}$ .*

## 1.2 Our Results

In this section, we state our main results for the erasure-resilient model of sublinear-time algorithms.

### 1.2.1 Testing Connectedness

The problem of testing connectedness in the *general graph model* (that we further generalize to the erasure-resilient setting) was studied by Parnas and Ron [23]. The results on this fundamental problem are described in Section 10.2.1 in [14]. The best tester for this problem to date, due to [5], had query complexity  $O(\frac{1}{(\varepsilon \bar{d})^2})$ .

We give two erasure-resilient testers for connectedness: one for small values of  $\alpha$  and another for intermediate values of  $\alpha$ . Both testers work for all<sup>3</sup> values of the proximity parameter,  $\varepsilon$ . We first give a tester that works for all  $\alpha < \varepsilon/2$ . (This tester is presented in Section 2.1.)

► **Theorem 1.4.** *There exists an  $\alpha$ -erasure-resilient  $\varepsilon$ -tester for connectedness of graphs with the average degree  $\bar{d}$  that has  $O(\min\{\frac{1}{((\varepsilon-2\alpha)\bar{d})^2}, \frac{1}{\varepsilon-2\alpha} \log \frac{1}{(\varepsilon-2\alpha)\bar{d}}\})$  query and time complexity and works for every  $\varepsilon \in (0, 2/\bar{d})$  and  $\alpha \in [0, \varepsilon/2)$ . The tester has 1-sided error. When the average degree  $\bar{d}$  of the input graph is unknown,  $\alpha$ -erasure-resilient  $\varepsilon$ -testing of connectedness (with 1-sided error) has query and time complexity  $O(\frac{1}{\varepsilon-2\alpha} \log \frac{1}{\varepsilon-2\alpha})$ .*

<sup>3</sup> For  $\varepsilon \geq 2/\bar{d}$ , we have  $\varepsilon m \geq n$ . Then testing connectedness is trivial, since every graph can be made connected by adding at most  $n - 1$  edges.

Importantly, when the input adjacency lists have no erasures (i.e., when  $\alpha = 0$ ), our tester has better query complexity than the previously known best (standard) tester for connectedness, which was due to [5]. We present a standalone algorithm for this important special case in the full version of this article [20]. By substituting  $\alpha = 0$  in Theorem 1.4, we get  $O(\min\{\frac{1}{(\varepsilon\bar{d})^2}, \frac{1}{\varepsilon} \log \frac{1}{\varepsilon\bar{d}}\})$  query complexity for the case when  $\bar{d}$  is known and  $O(\frac{1}{\varepsilon} \log \frac{1}{\varepsilon})$  query complexity when  $\bar{d}$  is unknown. For the case with no erasures, the improvement in query complexity as a function of  $\varepsilon$  is from  $O(\frac{1}{\varepsilon^2})$  to  $O(\frac{1}{\varepsilon} \log \frac{1}{\varepsilon})$ . The latter is optimal, as evidenced by an  $\Omega(\frac{1}{\varepsilon} \log \frac{1}{\varepsilon})$  lower bound for testing connectedness of graphs of degree 2 in [21]. We note that Berman et al. [5] already proved that testing connectedness of graphs (with no erasures) in the bounded degree graph model of [16] has query complexity  $O(\frac{1}{\varepsilon} \log \frac{1}{\varepsilon D})$  where  $D$  denotes the degree bound. Our result shows that the same query complexity (with  $D$  replaced by  $\bar{d}$ ) is attainable in the general graph model.

Our first tester looks for small connected components that do not have any erasures. When  $\alpha \in [\varepsilon/2, \varepsilon)$ , some  $\alpha$ -erased graphs that are  $\varepsilon$ -far from connected may not have any connected component that is free of erasures. Consequently, our first tester fails to reject such graphs. We give a different algorithm (presented in Section 2.2) which works by looking for a subset of vertices that has at most one erasure and gets completed to a unique connected component in every completion of the partially erased graph. (In the beginning of Section 2.2, we give an explanation, illustrated by Figure 1, of why two erasures in a witness may render it not detectable from a local view obtained by a sublinear algorithm.)

► **Theorem 1.5.** *There exists an  $\alpha$ -erasure-resilient  $\varepsilon$ -tester for connectedness of graphs with the average degree  $\bar{d}$  that has  $O(\frac{1}{(\varepsilon-\alpha)^2 \cdot \bar{d}} \cdot \min\{\frac{1}{(\varepsilon-\alpha) \cdot \bar{d}}, 1\})$  query and time complexity and works for every  $\varepsilon \in (0, 2/\bar{d})$  and  $\alpha \in [0, \varepsilon)$ . The tester has 1-sided error.*

Finally, we show that when  $\alpha \geq \varepsilon$ , the task of  $\alpha$ -erasure-resilient  $\varepsilon$ -testing of connectedness requires examining a linear portion of the graph representation. That is, we discover a phase transition in the complexity of this problem when the fraction of erasures  $\alpha$  reaches the proximity parameter  $\varepsilon$ .

► **Theorem 1.6.** *For all  $\varepsilon \in (0, 1/7]$ , every  $\varepsilon$ -erasure-resilient  $\varepsilon$ -tester for connectedness that makes only degree and neighbor queries requires a number of queries linear in the size of the graph representation.*

To prove this theorem, we construct (in Section 2.3) a family of partially erased graphs for which it is hard to distinguish connected graphs from graphs that are far from connected. The average degree of the graphs in our constructions is constant. So, the lower bound for this graph family is  $\Omega(n) = \Omega(m)$ .

## 1.2.2 Estimating the Average Degree

In Section 3.1, we give an erasure-resilient algorithm for estimating the average degree by generalizing the algorithm of Eden et al. [9, 10] to work for the case with erasures.

► **Theorem 1.7.** *Let  $\alpha \in [0, 1]$  and  $\varepsilon \in (0, 1/2)$ . There exists an algorithm that makes  $O(\sqrt{n} \cdot \log \log n \cdot \text{poly}(1/\varepsilon))$  degree queries and uniformly random neighbor queries to an  $\alpha$ -erased input graph of average degree  $\bar{d} \geq 1$  and outputs, with probability at least  $2/3$ , an estimate  $\tilde{d}$  satisfying  $(1 - \varepsilon) \cdot \bar{d} < \tilde{d} < (1 + 2 \min(\alpha, \frac{1}{2}) + \varepsilon) \cdot \bar{d}$ . The running time of the algorithm is the same as its query complexity.*

For graphs with no erasures, a good estimate of the number of edges gives a good estimate of the average degree. Feige’s algorithm [13] (that has access only to degree queries) counts some edges twice and gets an estimate of the average degree that is within a factor of  $2 + \varepsilon$ . Goldreich and Ron [17] and Eden et al. [9, 10] avoid the issue of double-counting by ranking vertices according to their degrees and estimating, within a factor of  $1 + \varepsilon$ , the number of edges going from lower-ranked to higher-ranked vertices. These algorithms use degree queries and uniformly random neighbor queries. Having erasures in the adjacency lists is, in a rough sense, equivalent to not having access to *some* of the neighbor queries. This results in the additional  $2\alpha$  error term in the approximation guarantee. Consequently, when the fraction of erasures approaches  $1/2$ , all the “relevant” entries in the adjacency lists of the input graph could be erased, and we enter the regime of having access only to degree queries.

In Section 3.2, we show that, for any fraction  $\alpha \in (0, 1]$ , estimating the average degree of an  $\alpha$ -erased graph to within a factor of  $(1 + \alpha)$  requires  $\Omega(n)$  queries. In other words, the approximation ratio of our erasure-resilient algorithm for estimating the average degree cannot be improved significantly.

► **Theorem 1.8.** *Let  $\alpha \in (0, 1]$  be rational. For all  $\gamma < \alpha$ , at least  $\Omega(n)$  queries are necessary for every algorithm that makes degree and neighbor queries to an  $\alpha$ -erased graph with the average degree  $\bar{d}$  and outputs, with probability at least  $2/3$ , an estimate  $\tilde{d} \in [\bar{d}, (1 + \gamma)\bar{d}]$ .*

### 1.3 Research Directions and Further Observations

There are numerous research questions that arise from our work. In Section 4, we discuss some of them and also give additional observations about variants of our model. We mention open questions about another (weaker) threshold in erasure-resilient testing of connectedness, about erasure-resilient testing of monotone graph properties, about the relationship between testing with erasures and testing with errors, and about the variant of our model that allows only symmetric erasures. We show that some of the questions we discuss are open in our model, but easy in the bounded-degree version of our model.

### 1.4 Related Work

Erasure-resilient sublinear-time algorithms, in the context of testing properties of functions, were first investigated by Dixit et al. [7], and further studied by Raskhodnikova et al. [25], Pallavoor et al. [22], and Ben-Eliezer et al. [3].

Property testing in the general graph model was first studied by Parnas and Ron [23], who considered a relaxed version of the problem of testing whether the input graph has small diameter. Kaufman et al. [19] studied the problem of testing bipartiteness in the general graph model and obtained tight upper and lower bounds on its complexity.

Sublinear-time algorithms for estimating various graph parameters have also received significant attention. There are sublinear-time algorithms for estimating the weight of a minimum weight spanning tree [6], the number of connected components [6, 4], the average degree [13, 17], the average pairwise distance [17], moments of the degree distribution [18, 9], and subgraph counts [18, 8, 11, 12, 1, 2].



## 2 Erasure-Resilient Testing of Connectedness

In this section, we present our results on erasure-resilient testing of connectedness in graphs.

### 2.1 An Erasure-Resilient Connectedness Tester for $\alpha < \varepsilon/2$

In this section, we present our connectedness tester for small  $\alpha$  and prove Theorem 1.4. The tester looks for witnesses to disconnectedness in the form of connected components with no erasures. It repeatedly performs a breadth first search (BFS) from a random vertex until it finds a witness to disconnectedness or exceeds a specified query budget.

A simple counting argument shows that if a partially erased graph is far from connected then it has many small witnesses to disconnectedness. Moreover, the size of the average witness among them is at most some bound  $b$  (that we calculate later). Our tester uses BFS to detect a witness to disconnectedness of size at most  $b$ .

The best tester for connectedness to date, by Berman et al. [5], uses a technique called the *work investment strategy*. Specifically, their algorithm repeatedly samples a uniformly random vertex  $v$ , guesses the size of the witness to disconnectedness  $C_{(v)}$  containing  $v$ , and then performs a BFS from  $v$  for  $|C_{(v)}|^2$  queries. Clearly,  $|C_{(v)}|^2$  queries are enough to detect  $C_{(v)}$ . Using the fact that the expected size of a witness is  $b$ , they argue that their algorithm has complexity  $O(b^2)$ .

The new idea in our connectedness tester is to perform the BFS from a uniformly random vertex  $v$  for  $|C_{(v)}| \cdot \deg(v)/2$  queries. The expected value of the latter quantity is bounded by  $E_{(v)}$ , where  $E_{(v)}$  denotes the number of edges in the witness containing  $v$ , and the expectation is over the choice of a uniformly random vertex from  $C_{(v)}$ . That is, in expectation, the number of queries that we *invest* into the BFS from  $v$  is enough to detect  $C_{(v)}$ . We show that, overall, the expected complexity of this algorithm is  $\tilde{O}(b \cdot \bar{d})$ , which is smaller than  $O(b^2)$  when  $b > \bar{d}$ .

Our erasure-resilient tester is Algorithm 1, with a small standard modification to ensure that the stated complexity bounds hold in the worst case (not just in expectation). It is obtained by running the algorithm of Berman et al. (generalized to handle erasures) when  $b < \bar{d}$  and running the above algorithm otherwise.

Before stating the algorithm, we formalize the notion of the witness to disconnectedness and argue that partially erased graphs that are far from being connected have many witnesses to disconnectedness.

► **Definition 2.1** (Witness to disconnectedness). *A set  $C$  of vertices is a witness to disconnectedness in a partially erased graph  $G$  if the adjacency lists of vertices in  $C$  have no erasures, and  $C$  forms a connected component in every completion of  $G$ .*

► **Observation 2.2.** *Let  $\varepsilon \in (0, 2/\bar{d})$  and  $G'$  be an  $m$ -edge graph (with no erasures) that is  $\varepsilon$ -far from connected. Then  $G'$  has at least  $\varepsilon m + 1$  connected components.*

Next, in Claim 2.3, we argue that if the fraction of erasures is *small*, *many* of the connected components present in a completion  $G'$  are also present as witnesses to disconnectedness in  $G$ .

▷ **Claim 2.3.** *Let  $\varepsilon \in (0, 2/\bar{d})$  and  $\alpha \in [0, \varepsilon/2)$ . The number of witnesses to disconnectedness in an  $\alpha$ -erased graph  $G$  that is  $\varepsilon$ -far from connected is at least  $(\varepsilon - 2\alpha)m$ .*

*Proof.* By Observation 2.2, every completion  $G'$  of  $G$  has at least  $\varepsilon m + 1$  connected components. The number of connected components in  $G'$  with at least one erased entry in the union of its adjacency lists (with respect to  $G$ ) is at most  $2\alpha m$ . Hence, the number of connected components in  $G'$  that do not have any erased entry in the union of its adjacency lists (with respect to  $G$ ) is at least  $\varepsilon m - 2\alpha m = (\varepsilon - 2\alpha)m$ . The claim follows. ◁



Let  $b = 2/((\varepsilon - 2\alpha) \cdot \bar{d})$ . By Claim 2.3, the size of the average witness to disconnectedness is at most  $b$ . Now we are ready to state Algorithm 1.

■ **Algorithm 1** Erasure-Resilient Connectedness Tester for  $\alpha < \varepsilon/2$ .

---

**input** : The average degree  $\bar{d}$ , parameters  $\varepsilon \in (0, 2/\bar{d})$ ,  $\alpha \in [0, \varepsilon/2)$ ; query access to an  $\alpha$ -erased graph  $G$ .

- 1 Let  $b \leftarrow 2/((\varepsilon - 2\alpha) \cdot \bar{d})$ . // the average size of a witness is at most  $b$
- 2 **for**  $i \in [\lceil \log(4b) \rceil]$  **do**
- 3    **repeat**  $\lceil \frac{4b \ln 6}{2^i} \rceil$  **times**
- 4     Sample a vertex  $v$  uniformly and independently at random.
- 5     **if**  $b \leq \bar{d} \log b$  **then**
- 6       Run a BFS from  $v$  until it encounters an erased entry or  $(2^i + 1)$  vertices.
- 7       **else**
- 8          Query  $\deg(v)$ ;
- 9          Run a BFS from  $v$  until it encounters an erased entry or  $(2^{i-1} \cdot \deg(v) + 1)$  edges.
- 10      **if** the BFS explored an entire connected component and didn't encounter an erasure **then reject**.
- 11 **Accept**.

---

Clearly, Algorithm 1 accepts all connected partially erased graphs.

► **Lemma 2.4.** *Let  $\varepsilon \in (0, 2/\bar{d})$  and  $\alpha \in [0, \varepsilon/2)$ . Let  $G$  be an  $\alpha$ -erased graph that is  $\varepsilon$ -far from connected. Then Algorithm 1 rejects  $G$  with probability at least  $5/6$ .*

**Proof.** Let  $V$  be the vertex set of  $G$ . We start by defining the quality of a vertex  $v \in V$ . The definition is different for the two cases, corresponding to the two stopping conditions Algorithm 1 uses for BFS. First, we consider the case when  $b \leq \bar{d} \cdot \log b$ , that is, when Algorithm 1 runs the version of BFS specified in Step 6.

► **Definition 2.5** (Quality of a vertex when  $b \leq \bar{d} \cdot \log b$ ). *The quality of a vertex  $v$ , denoted  $q(v)$ , is defined as follows. If  $v$  belongs to a witness to disconnectedness in  $G$  then  $q(v) = 1/|C_{(v)}|$ , where  $C_{(v)}$  denotes the witness to disconnectedness that  $v$  belongs to. Otherwise,  $q(v) = 0$ .*

The important feature of  $q(v)$  is that, for a witness  $C$  to disconnectedness,  $\sum_{v \in C} q(v) = 1$ .

Next, we define the quality of a vertex for the case when  $b > \bar{d} \cdot \log b$ , that is, when Algorithm 1 runs the version of BFS specified in Step 8.

► **Definition 2.6** (Quality of a vertex when  $b > \bar{d} \cdot \log b$ ). *Fix a completion  $G'$  of  $G$ . For a vertex  $v \in V$ , let  $C_{(v)}$  denote the connected component (in  $G'$ ) containing  $v$ , and let  $E_{(v)}$  denote the number of edges in  $C_{(v)}$ . The quality of a vertex  $v$ , denoted  $q(v)$ , is defined as*

$$q(v) = \begin{cases} 0 & \text{if } C_{(v)} \text{ contains at least one erased entry in } G, \\ \frac{\deg(v)}{2E_{(v)}} & \text{if } E_{(v)} > 0, \\ 1 & \text{if } E_{(v)} = 0. \end{cases}$$

Like for  $q(v)$  from Definition 2.5, for a witness  $C$  to disconnectedness,  $\sum_{v \in C} q(v) = 1$ .

The rest of the proof of Lemma 2.4 is the same for both cases. We analyze the expected quality of a uniformly random vertex  $v \in V$ . Since  $\sum_{v \in C} q(v) = 1$ , by Claim 2.3,

$$\mathbb{E}_{v \in V}[q(v)] = \frac{1}{n} \sum_{v \in V} q(v) = \frac{1}{n} \sum_{\substack{C: C \text{ is a witness} \\ \text{to disconnectedness}}} 1 \geq \frac{(\varepsilon - 2\alpha)m}{n} = \frac{1}{b}.$$

Finally, we apply the following work investment strategy lemma [5, Lemma 2.5].

► **Lemma 2.7** ([5]). *Let  $X$  be a random variable that takes values in  $[0, 1]$ . Suppose  $\mathbb{E}[X] \geq \beta$ , and let  $t = \lceil \log(4/\beta) \rceil$ . For all  $i \in [t]$ , let  $p_i = \Pr[X \geq 2^{-i}]$  and  $k_i = \frac{4 \ln 6}{2^i \beta}$ . Then  $\prod_{i=1}^t (1 - p_i)^{k_i} \leq \frac{1}{6}$ .*

We apply Lemma 2.7 with  $X$  equal to  $q(v)$  for a uniformly random  $v \in V$ . Set  $\beta = 1/b$  and  $t = \lceil \log(4/\beta) \rceil$ . For  $i \in [t]$ , set  $p_i$  to be the probability that a vertex  $v$  sampled uniformly at random belongs to a witness to disconnectedness of  $G$  that has at most (i)  $2^i$  vertices, when  $b \leq \bar{d} \cdot \log b$ ; (ii)  $2^{i-1} \cdot \deg(v)$  edges, otherwise. That is,  $p_i = \Pr[X \geq 2^{-i}]$ . Similarly, for  $i \in [t]$ , let  $k_i = \frac{4 \ln 6}{2^i \beta}$ . Then the probability that Step 9 of the tester does not reject is  $\prod_{i=1}^t (1 - p_i)^{k_i}$ . By Lemma 2.7, this step rejects with probability at least  $5/6$ . ◀

**Proof of Theorem 1.4.** We start by analyzing the query and time complexity of Algorithm 1.

**Case 1:** When  $b \leq \bar{d} \cdot \log b$ , the query and time complexity of Algorithm 1 is

$$\sum_{i \in [\lceil \log(4b) \rceil]} \left\lceil \frac{4b \ln 6}{2^i} \right\rceil \cdot 2^{2i} = O(b^2) = O(\min\{b^2, b\bar{d} \cdot \log b\}).$$

**Case 2:** When  $b > \bar{d} \cdot \log b$ , the expected query and time complexity of Algorithm 1 is

$$\sum_{i \in [\lceil \log 4b \rceil]} \left\lceil \frac{4b \ln 6}{2^i} \right\rceil \cdot 2^i \cdot \mathbb{E}_{s \in V}[\deg(s)] = O(b\bar{d} \log b) = O(\min\{b^2, b\bar{d} \cdot \log b\}).$$

Substituting the value of  $b$ , we get

$$O(\min\{b^2, b\bar{d} \cdot \log b\}) = O\left(\min\left\{\frac{1}{((\varepsilon - 2\alpha)\bar{d})^2}, \frac{1}{\varepsilon - 2\alpha} \log \frac{1}{(\varepsilon - 2\alpha)\bar{d}}\right\}\right).$$

The final tester is obtained by running Algorithm 1 and then aborting and accepting if the number of queries exceeds six times its expectation. The final tester then has the query complexity and the running time stated in Theorem 1.4.

The final tester never rejects a connected partially erased graph. However, a partially erased graph that is  $\varepsilon$ -far from connected can get accepted incorrectly if Algorithm 1 accepts it or if the final algorithm aborts. The probability of the former event is at most  $1/6$ , by Lemma 2.4. The probability of aborting is also at most  $1/6$ , by Markov's inequality. By a union bound, the final algorithm accepts incorrectly with probability at most  $1/3$ , completing the proof of the theorem for the case when  $\bar{d}$  is given to the algorithm.

We can adjust the algorithm to work without access to the average degree at a small cost in query and time complexity. The details appear in the full version [20]. ◀

## 2.2 Our Erasure-Resilient Connectedness Tester for $\alpha \in [\varepsilon/2, \varepsilon]$

In this section, we prove Theorem 1.5. We describe and analyze a 1-sided error  $\alpha$ -erasure-resilient  $\varepsilon$ -tester for connectedness that can work with more erasures in the input graph than Algorithm 1 can handle. Specifically, the tester works for all  $\alpha < \varepsilon$ . However, it has better performance than Algorithm 1 only for  $\alpha \in [\varepsilon/2, \varepsilon]$ .



■ **Figure 1** An example of a component with two erasures, where a BFS from any vertex fails to detect that this component is disconnected from the rest of the graph.



■ **Figure 2** An example of a generalized witness to disconnectedness, where only a BFS from  $v_1$  (but not from any other vertex) detects the generalized witness.

A dotted line represents an erasure in the adjacency list of the corresponding vertex. An arrow pointing from a vertex  $a$  in the direction of a vertex  $b$  represents that  $b \in \text{Adj}(a)$ , but  $a \notin \text{Adj}(b)$ .

When  $\alpha > \varepsilon/2$ , an  $\alpha$ -erased graph that is  $\varepsilon$ -far from being connected may not contain any witnesses to disconnectedness as defined in Section 2.1. Specifically, every set  $C$  of nodes that gets completed to a connected component could have an erasure in the union of the adjacency lists of the nodes in  $C$ . To get around this issue, our tester looks for a *generalized witness to disconnectedness*, which is, intuitively, a connected component with at most one erasure. Observe that a component with two erasures could have a unique completion, but impossible to certify as a separate connected component from the local view from any of its vertices. Figure 1 shows an example of a small component, where a BFS from any vertex will be unable to certify that the graph is disconnected.

Our tester repeatedly performs a BFS from a random vertex until it detects a generalized witness to disconnectedness, or exceeds a specified query budget. We show, by a counting argument, that every partially erased graph that is far from connected has several *small* generalized witnesses to disconnectedness. The correctness of the tester is ensured by the observation that each such witness  $C$  contains at least one vertex from which all the other vertices in  $C$  are reachable. (It is possible to have *exactly* one vertex in  $C$  from which all the other vertices are reachable. Figure 2 shows an example of a connected component, where a BFS can detect the generalized witness to disconnectedness only if started at vertex  $v_1$ , but will fail to do so from all other vertices.)

Before we state our tester, we formalize the notion of generalized witnesses.

► **Definition 2.8** (Generalized witness to disconnectedness). *Given a partially erased graph  $G$  over a vertex set  $V$ , a set  $C \subset V$  is a generalized witness to disconnectedness of  $G$  if*

1. *there is at most one erased entry ( $\perp$ ) in  $\bigcup_{v \in C} \text{Adj}(v)$ ,*
2. *every nonerased entry in  $\bigcup_{v \in C} \text{Adj}(v)$  is a vertex from  $C$ ,*
3. *if  $\perp \in \text{Adj}(u)$  for some  $u \in C$  then  $u \in \text{Adj}(v)$  but  $v \notin \text{Adj}(u)$  for some  $v \in C$ ; moreover, each node in  $C$  is reachable via a BFS from  $v$ .*

Definition 2.8 implies that the only erasure, if any, in the union of the adjacency lists of the nodes in  $C$  is part of a half-erased edge within  $C$ , and that  $C$  forms a connected component in every completion of  $G$ .

Let  $b = 4/((\varepsilon - \alpha)\bar{d})$ . Our tester is presented in Algorithm 2. In the rest of the section, we analyze the correctness and complexity of the tester.

► **Definition 2.9** (Small and big sets). *Let  $G$  be a partially erased graph and let  $\varepsilon^* \in (0, 2/\bar{d})$  be a parameter. The representation length of a set  $C$  of nodes is the sum of lengths of the adjacency lists of nodes in  $C$ . The set  $C$  is  $\varepsilon^*$ -small if either*

- $\varepsilon^* \geq 4/\bar{d}^2$  and  $C$  contains at most  $4/(\varepsilon^* \cdot \bar{d})$  vertices, or
- $\varepsilon^* < 4/\bar{d}^2$  and  $C$  has representation length at most  $4/\varepsilon^*$ .

*The set  $C$  is  $\varepsilon^*$ -big otherwise.*

Claim 2.10 shows that a partially erased graph that is far from connected has sufficiently many small generalized witnesses to disconnectedness.

---

**Algorithm 2** Erasure-Resilient Connectedness Tester for  $\alpha \in [\varepsilon/2, \varepsilon)$ .

---

**input** : The average degree  $\bar{d}$ , parameters  $\varepsilon \in (0, 2/\bar{d})$ ,  $\alpha \in [0, \varepsilon)$ ; query access to an  $\alpha$ -erased graph  $G$ .

- 1 Let  $b \leftarrow 4/((\varepsilon - \alpha)\bar{d})$ .
- 2 **repeat**  $\lceil b \ln 3 \rceil$  **times**
- 3   Sample a vertex  $s$  uniformly and independently at random.
- 4   Run a BFS starting from  $s$  using at most  $\min\{b^2, b \cdot \bar{d}\}$  neighbor queries.
- 5   **if** *Step 4 detected a generalized witness to disconnectedness* **then**
- 6     **Reject**.
- 7 **Accept**.

---

▷ **Claim 2.10.** Let  $\varepsilon \in (0, 2/\bar{d})$ ,  $\alpha \in [0, \varepsilon)$ . Let  $G$  be an  $\alpha$ -erased graph that is  $\varepsilon$ -far from connected. The number of  $(\varepsilon - \alpha)$ -small generalized witnesses to disconnectedness of  $G$  is at least  $(\varepsilon - \alpha)m/2$ .

*Proof.* We first argue that there are many small connected components in every completion  $G'$  of  $G$  and then prove that many of these are generalized witnesses in  $G$ .

Consider a completion  $G'$  of  $G$ . If  $\varepsilon - \alpha \geq 4/\bar{d}^2$ , the number of  $(\varepsilon - \alpha)$ -big connected components in  $G'$  is at most  $n/b = (\varepsilon - \alpha)m/2$ . If  $\varepsilon - \alpha < 4/\bar{d}^2$ , the number of  $(\varepsilon - \alpha)$ -big connected components in  $G'$  is at most  $2m/(b \cdot \bar{d}) = (\varepsilon - \alpha)m/2$ , since the representation length of the vertex set  $V$  of  $G$  is  $2m$ . By Observation 2.2, the total number of connected components in  $G'$  is at least  $\varepsilon m + 1$ . Hence, the number of  $(\varepsilon - \alpha)$ -small connected components in  $G'$  is at least  $(\varepsilon + \alpha)m/2$ .

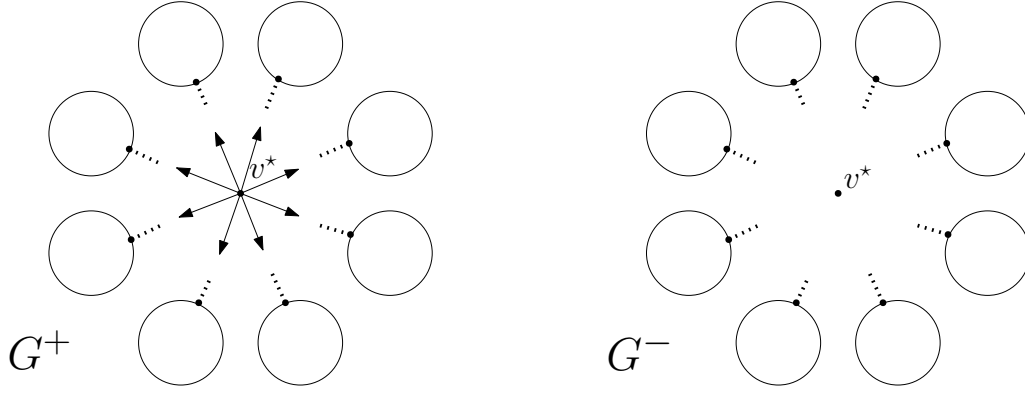
Let  $C \subset V$  denote the set of vertices corresponding to an  $(\varepsilon - \alpha)$ -small connected component in  $G'$ . If  $\bigcup_{v \in C} \text{Adj}(v)$  has no erasures, then  $C$  is a generalized witness to disconnectedness of  $G$ . Next, assume that  $\bigcup_{v \in C} \text{Adj}(v)$  has exactly one erasure. We show that the set  $C$  is a generalized witness to disconnectedness of  $G$ . Condition 1 is satisfied by definition. Condition 2 is true since  $C$  forms a connected component in  $G'$ . To see that Condition 3 holds, let  $u \in C$  be the vertex with  $\perp \in \text{Adj}(u)$ . Since  $C$  is a connected component in  $G'$ , this erased entry was completed with the label of another vertex  $v \in C$ . Moreover, every vertex in  $C$  is reachable by a BFS from  $v$ , since  $C$  forms a connected component in  $G'$ , and the erased entry is not needed for these searches because it would lead back to  $v$ . Therefore,  $C$  is a generalized witness to disconnectedness of  $G$  if  $\bigcup_{v \in C} \text{Adj}(v)$  has exactly one erasure.

Among the  $(\varepsilon - \alpha)$ -small connected components in  $G'$ , at most  $\alpha m$  have at least 2 erased entries in the union of their adjacency lists. Hence, the number of  $(\varepsilon - \alpha)$ -small generalized witnesses to disconnectedness of  $G$  is at least  $((\varepsilon + \alpha)m/2) - \alpha m = (\varepsilon - \alpha)m/2$ . ◁

Lemma 2.11 below implies Theorem 1.5.

► **Lemma 2.11.** For every  $\varepsilon \in (0, 2/\bar{d})$  and  $\alpha \in [0, \varepsilon)$ , Algorithm 2 is an  $\alpha$ -erasure-resilient  $\varepsilon$ -tester for connectedness of graphs with the average degree  $\bar{d}$ . It has  $O(b^2 \bar{d} \cdot \min\{b/\bar{d}, 1\})$  query and time complexity.

**Proof.** Consider an  $\alpha$ -erased graph  $G$  over a vertex set  $V$ . Assume that  $G$  is connected, that is, there exists a connected completion  $G'$  of  $G$ . Consider an arbitrary  $C \subset V$ . There exist vertices  $u \in C$  and  $v \in V \setminus C$  such that  $\text{Adj}(u)$  in  $G'$  contains  $v$ . Hence,  $C$  is not a generalized witness to disconnectedness of  $G$ . Therefore, the tester accepts  $G$ .



■ **Figure 3** The partially erased graphs  $G^+$  and  $G^-$  described in the proof of Theorem 1.6. The dotted lines represent erased entries in the adjacency lists of the corresponding vertices. In  $G^+$ , the directed edges from  $v^*$  point to the vertices in its adjacency list. The circles represent cycles.

Next, assume that  $G$  is  $\varepsilon$ -far from connected. Let  $\mathcal{W}$  denote the family of all  $(\varepsilon - \alpha)$ -small generalized witnesses to disconnectedness of  $G$ . Let  $C \subset V$  be an element of  $\mathcal{W}$ . If  $\varepsilon - \alpha \geq 4/\bar{d}^2$ , the representation length of  $C$  is at most  $b^2 \leq b \cdot \bar{d}$ . If  $\varepsilon - \alpha < 4/\bar{d}^2$ , the representation length of  $C$  is at most  $b \cdot \bar{d} < b^2$ . Hence, the representation length of  $C$  is at most  $\min\{b^2, b \cdot \bar{d}\}$ . If  $\bigcup_{v \in C} \text{Adj}(v)$  has no erasures then every vertex in  $C$  is reachable from every other vertex in  $C$ . Otherwise, the vertex  $v$  in Condition 3 of Definition 2.8 is such a vertex. If Algorithm 2 performs a BFS from  $v$ , it will detect a generalized witness to disconnectedness after at most  $\min\{b^2, b \cdot \bar{d}\}$  queries and reject. Since  $|\mathcal{W}| \geq (\varepsilon - \alpha)m/2$  and each generalized witness in  $\mathcal{W}$  has at least one vertex from which the generalized witness is detectable by a BFS, a single iteration of Algorithm 2 rejects with probability at least  $|\mathcal{W}|/n = 1/b$ . Hence, Algorithm 2 rejects with probability at least  $1 - (1 - (1/b))^{\lceil b \ln 3 \rceil} \geq 1 - \exp(-\ln 3) = 2/3$ .

Step 4 of Algorithm 2 makes at most  $\min\{b^2, b\bar{d}\}$  queries. Thus, the query complexity of Algorithm 2 is  $O(b \cdot \min\{b^2, b\bar{d}\})$ , which simplifies to the claimed expression. Checking (in Step 5) whether a set  $C$  is a generalized witness to disconnectedness can be done with a constant number of passes over the adjacency lists of vertices in  $C$ . Since the algorithm queried all entries in them, its running time is asymptotically equal to its query complexity. ◀

### 2.3 A Lower Bound for Erasure-Resilient Connectedness Testing

In this section, we prove Theorem 1.6. We note that hard graphs in our construction have constant average degree. That is, for those graphs, our lower bound is  $\Omega(n) = \Omega(m)$ .

**Proof of Theorem 1.6.** We apply Yao's minimax principle, as stated in [26]. Specifically, we construct distributions  $\mathcal{D}^+$  and  $\mathcal{D}^-$ , the former over connected graphs and the latter over graphs that are  $\varepsilon$ -far from connected, such that every deterministic  $\varepsilon$ -erasure-resilient  $\varepsilon$ -tester for connectedness makes  $\Omega(m)$  queries to distinguish the two distributions.

Without loss of generality, assume that  $t = (1 - \varepsilon)/(2\varepsilon)$  is an integer. Observe that  $t \geq 3$  as  $\varepsilon \leq 1/7$ . Let  $k$  be an even number and  $n = kt + 1$ . We first construct two partially erased  $n$ -node graphs  $G^+$  and  $G^-$ , depicted in Figure 3. The vertices of  $G^+$  are partitioned into  $k + 1$  sets. Each of the first  $k$  sets induces a  $t$ -node cycle. Exactly one node in each cycle has degree 3 and has an erasure in its adjacency list, in addition to its two neighbors on the cycle. The last set contains a single node  $v^*$  of degree  $k$ . Its adjacency list contains the labels of the degree-3 vertices in the cycles. The graph  $G^-$  is the same as  $G^+$ , except that in  $G^-$ , we have that  $\text{Adj}(v^*)$  is empty, that is,  $v^*$  is isolated.

We can obtain a connected completion of  $G^+$  by connecting the vertex  $v^*$  to all the degree-3 vertices. In contrast, at least  $k/2$  edges need to be added to every completion of  $G^-$  to make it connected. Hence, the distance from  $G^-$  to connectedness is  $(k/2)/(kt+k/2) = 1/(2t+1) = \varepsilon$ .

The fraction of erased entries in the adjacency lists of  $G^+$  and  $G^-$  are  $1/(2t+2)$  and  $1/(2t+1)$ , respectively. That is,  $G^+$  and  $G^-$  are both  $\alpha$ -erased graphs for  $\alpha = 1/(2t+1) = \varepsilon$ .

The distributions  $\mathcal{D}^+$  and  $\mathcal{D}^-$  are uniform over the sets of all partially erased graphs isomorphic to  $G^+$  and  $G^-$ , respectively. Each partially erased graph sampled from  $\mathcal{D}^+$  is connected. Each partially erased graph sampled from  $\mathcal{D}^-$  is  $\varepsilon$ -far from connected.

▷ **Claim 2.12.** Every deterministic algorithm  $A$  has to make  $\Omega(n)$  queries to distinguish  $\mathcal{D}^+$  and  $\mathcal{D}^-$  with probability at least  $2/3$ .

*Proof.* Let  $q$  denote the number of queries made by  $A$  and assume  $q \leq n/6$ . In this proof, we use  $v^*$  as a shorthand for the vertex from the singleton set in the construction of  $\mathcal{D}^+$  and  $\mathcal{D}^-$ , as opposed to the label of that vertex. Since  $\mathcal{D}^+$  and  $\mathcal{D}^-$  differ only on  $v^*$ , it is important to understand when  $A$  gets any information about  $v^*$ .

► **Definition 2.13 (Node status).** *Given a sequence of queries made by  $A$  and answers it has received so far, a node  $v$  is known if it has been queried (via a degree or neighbor query) or received as an answer to a (neighbor) query; otherwise, it is unknown.*

The node  $v^*$  is *unknown* before  $A$  makes its first query. Since  $v^*$  cannot be received as an answer to a query for the graphs in the support of  $\mathcal{D}^+$  and  $\mathcal{D}^-$ , it can become *known* only if  $A$  queries an *unknown* node that happens to be  $v^*$ . At most two new nodes become *known* per query. So, the probability (over the distribution  $\mathcal{D}^+$  or  $\mathcal{D}^-$ ) that a specific *unknown* node queried by  $A$  turns out to be  $v^*$  is at most  $1/(n-2q)$ . Let  $p$  denote the probability that  $v^*$  becomes *known* by the end of an execution of  $A$ . By a union bound over all queries made by  $A$ , we have,  $p \leq \frac{q}{n-2q} \leq \frac{n/6}{n-n/3} = \frac{1}{4}$ .

If  $v^*$  is *unknown* by the end of a particular execution then the view of the partially erased graph obtained by  $A$  in that execution arises with the same probability under  $\mathcal{D}^+$  and under  $\mathcal{D}^-$ . Such an execution of  $A$  can distinguish  $\mathcal{D}^+$  and  $\mathcal{D}^-$  with probability at most  $1/2$ . Therefore, the probability that  $A$  distinguishes  $\mathcal{D}^+$  and  $\mathcal{D}^-$  is at most  $p + (1-p) \cdot \frac{1}{2} = \frac{1}{2} + \frac{p}{2} < \frac{2}{3}$ . ◁

In our construction,  $m = \Theta(n)$ . Thus, every  $\varepsilon$ -erasure-resilient  $\varepsilon$ -tester for connectedness that uses only degree and neighbor queries must make  $\Omega(m)$  queries in the worst case over the input graph, completing the proof of Theorem 1.6. ◀

### 3 Estimating the Average Degree of a Graph

In this section, we present our results on erasure-resilient estimation of the average degree of graphs.

#### 3.1 An Algorithm for Estimating the Average Degree

In this section, we describe and analyze the algorithm (claimed in Theorem 1.7) for estimating the average degree of (or, equivalently, the number of edges in) a partially erased graph. Our algorithm is a generalization of the algorithm for counting the number of edges in graphs by Eden et al. [9, 10] to the case of partially erased graphs. We first give an algorithm (Algorithm 3) that takes a crude estimate of the average degree as input and outputs a more accurate estimate. Using a standard technique similar to the binary search, our final algorithm uses Algorithm 3 as a subroutine to gradually refine its estimate of the average degree. The final algorithm and the complete proof of Theorem 1.7 appear in the full version.

Algorithm 3, like the algorithm of Eden et al. [9, 10], works by empirically estimating a random variable whose expectation is close to the number of edges in the graph. We first rank vertices according to their degrees, breaking ties arbitrarily. Then we orient the nonerased edges of the graph from lower-ranked to higher-ranked endpoints. This orientation allows us to attribute each nonerased edge to its lower-ranked endpoint in order to avoid double-counting the edge. Since the number of edges between high-degree vertices is small, we ignore such edges. Algorithm 3 samples low-degree vertices uniformly at random and estimates, via sampling, the number of edges “credited” to them.

The crucial difference in the behavior of the algorithm in the case of partially erased graphs is the following. When we sample an erased entry from the adjacency list of a low-degree vertex  $u$ , we assume that it gets completed to a vertex ranked higher than  $u$  and, therefore, attribute the corresponding edge to  $u$ . Consequently, some erased edges get counted twice. This results in the additional term depending on the fraction of erasures in the approximation guarantee.

The ranking of (or the total ordering on) the vertices of a graph is defined below.

► **Definition 3.1** (Total ordering  $\prec$ ). *In a partially erased graph  $G$ , for any two vertices  $u, v$ , we write  $u \prec v$  if either  $\deg(u) < \deg(v)$ , or  $\deg(u) = \deg(v)$  and  $u$  is lexicographically smaller than  $v$ .*

■ **Algorithm 3** Erasure-Resilient Algorithm for Improving an Estimate of Average Degree.

---

**input** : Parameters  $\varepsilon \in (0, 1/2)$ ,  $\delta \in (0, 1/3)$ ; query access to a partially erased graph  $G$  on  $n$  nodes; a crude estimate  $\widehat{d}$  of the average degree of  $G$ .

- 1 Set  $s \leftarrow \left\lceil 660 \ln(2/\delta) \sqrt{\frac{n}{\varepsilon^5 \widehat{d}}} \right\rceil$ .
- 2 **for**  $i = 1$  **to**  $s$  **do**
- 3     Sample a node  $u$  from  $V$  uniformly at random and query its degree,  $\deg(u)$ .
- 4     Query a uniformly random entry from  $\text{Adj}(u)$  and let  $v$  be the answer.
- 5     If  $v \neq \perp$  then query its degree,  $\deg(v)$ .
- 6     **if**  $\deg(u) \leq 4\sqrt{n\widehat{d}/\varepsilon}$  **and** either  $v = \perp$  or  $u \prec v$  **then**
- 7          $\chi_i \leftarrow \deg(u)$
- 8     **else**
- 9          $\chi_i \leftarrow 0$
- 9 **return**  $\widetilde{d} = 2 \cdot \frac{1}{s} \sum_{i=1}^s \chi_i$ .

---

► **Lemma 3.2.** *Let  $G$  be an  $\alpha$ -erased  $n$ -node graph with the average degree  $\bar{d} \geq 1$ . Let  $\widehat{d}$  be a crude estimate of the average degree, given as an input to Algorithm 3. Then the output  $\widetilde{d}$  of Algorithm 3 satisfies the following:*

1. *If  $\widehat{d} \geq \frac{\bar{d}}{8}$  then, with probability at least  $3/4$ , we have  $\widetilde{d} \leq 8\bar{d}$ .*
2. *Furthermore, if  $\frac{\bar{d}}{8} \leq \widehat{d} \leq 8\bar{d}$  then with probability at least  $1 - \delta$ ,*

$$(1 - \varepsilon) \cdot \bar{d} < \widetilde{d} < (1 + \varepsilon + 2 \min(\alpha, \frac{1}{2})) \cdot \bar{d}.$$

*The query complexity of the algorithm is  $\Theta\left(\sqrt{\frac{n}{\varepsilon^5 \widehat{d}}} \cdot \log \frac{1}{\delta}\right)$ .*



**Proof.** The algorithm makes at most two degree queries and one neighbor query in each iteration, and it runs for  $\Theta\left(\sqrt{\frac{n}{\varepsilon^5 \cdot \bar{d}}} \cdot \log \frac{1}{\delta}\right)$  iterations. Hence, the bound on its query complexity is as claimed in the lemma.

To prove the guarantees on the output estimate  $\tilde{d}$ , we first show that for all  $i \in [s]$ , the expected value of  $\chi_i$  is a good estimate to the average degree of the partially erased graph, where  $s$  is the number of samples taken by Algorithm 3. We then apply Markov's inequality and Chernoff bound to prove parts 1 and 2 of the lemma, respectively. For all  $i \in [s]$ , the random variables  $\chi_i$  set by the algorithm are mutually independent and identically distributed. Hence, it suffices to bound  $\mathbb{E}[\chi_1]$ .

▷ **Claim 3.3.** If  $\hat{d} \geq \frac{\bar{d}}{8}$  then

$$\left(1 - \frac{\varepsilon}{2}\right) \cdot \frac{\bar{d}}{2} < \mathbb{E}[\chi_1] \leq \left(1 + 2 \min\left(\alpha, \frac{1}{2}\right)\right) \cdot \frac{\bar{d}}{2}.$$

**Proof.** Let  $m = n\bar{d}/2$  denote the total number of edges in the graph, and

$$\mathcal{H} = \left\{u \in V \mid \deg(u) > 4\sqrt{n\hat{d}/\varepsilon}\right\}$$

denote the set of high degree vertices. Let  $\hat{m} = n\hat{d}/2$  be the number of edges in the graph estimated from the input parameter  $\hat{d}$ . Since  $\hat{d} \geq \bar{d}/8$ , we have  $\hat{m} \geq m/8$ . Hence,

$$|\mathcal{H}| < \frac{2m}{4\sqrt{n\hat{d}/\varepsilon}} = \frac{m}{2\sqrt{2\hat{m}/\varepsilon}} \leq \frac{m}{\sqrt{m/\varepsilon}} = \sqrt{\varepsilon m}, \quad (1)$$

where the first inequality holds because the sum of degrees of high-degree vertices is at most  $2m$ , and the second inequality follows from  $\hat{m} \geq m/8$ .

The following quantity,  $d^+(u)$ , was defined in [10] for (standard) graphs. We extend their definition to partially erased graphs.

► **Definition 3.4.** For a vertex  $u$  in a partially erased graph  $G$ , let  $N(u)$  denote the set of (nonerased) neighbors present in  $\text{Adj}(u)$ . Let  $d^+(u) = |\{v \in N(u) \mid u \prec v\}|$  denote the number of nonerased neighbors of  $u$  that are higher than  $u$  w.r.t. the ordering on vertices (as in Definition 3.1).

Roughly,  $d^+(u)$  denotes the number of nonerased neighbors of  $u$  with the degree higher than that of  $u$ . The following fact is based on an observation by [10].

► **Fact 3.5.** For a partially erased graph  $G$  over a vertex set  $V$ , the sum  $\sum_{u \in V} d^+(u) \leq m$ . The inequality can be replaced with equality when  $G$  has no erasures.

The fact holds because each nonerased and half-erased edge in  $G$  is counted exactly once and at most once, respectively, in the sum  $\sum_{u \in V} d^+(u)$ .

Let  $u_1, u_2, \dots, u_{|\mathcal{H}|}$  be a labeling of the high degree vertices such that  $u_1 \prec u_2 \prec \dots \prec u_{|\mathcal{H}|}$ . For each  $j \in [|\mathcal{H}|]$ , observe that  $d^+(u_j) \leq |\mathcal{H}| - j$ , as  $d^+(u_j)$  is at most the number of vertices that are higher than  $u_j$  in the ordering. Hence,

$$\sum_{u \in \mathcal{H}} d^+(u) \leq \sum_{j=1}^{|\mathcal{H}|} (|\mathcal{H}| - j) = \sum_{k=0}^{|\mathcal{H}|-1} k < \frac{|\mathcal{H}|^2}{2} < \frac{\varepsilon m}{2}, \quad (2)$$

where the last inequality follows from (1).

Let  $d^\perp(u)$  denote the number of erased entries in  $\text{Adj}(u)$ . The expectation

$$\mathbb{E}[\chi_1] = \frac{1}{n} \sum_{u \in V \setminus \mathcal{H}} \frac{d^+(u) + d^\perp(u)}{\deg(u)} \cdot \deg(u) = \frac{1}{n} \sum_{u \in V \setminus \mathcal{H}} (d^+(u) + d^\perp(u)) \quad (3)$$

since the degree of the sampled vertex  $u$  is assigned to  $\chi_1$  if and only if

1.  $\deg(u) \leq 4\sqrt{n\widehat{d}}/\varepsilon$ , i.e.,  $u \in V \setminus \mathcal{H}$ ; and
2. the queried entry from  $\text{Adj}(u)$  is either a vertex  $v \succ u$  or  $\perp$ .

We now bound the quantity on the right hand side of (3) from below and above. Let  $G'$  be an arbitrary completion of  $G$ , and let  $d_{G'}^+(\cdot)$  denote the quantity defined in Definition 3.4 with respect to  $G'$  (instead of  $G$ ). For each  $u \in V$ , observe that  $d^+(u) + d^\perp(u) \geq d_{G'}^+(u)$ . Note that the upper bound in (2) still holds if we replace  $d^+(\cdot)$  with  $d_{G'}^+(\cdot)$ . Hence, by (3),

$$\mathbb{E}[\chi_1] \geq \frac{1}{n} \sum_{u \in V \setminus \mathcal{H}} d_{G'}^+(u) = \frac{1}{n} \left( m - \sum_{u \in \mathcal{H}} d_{G'}^+(u) \right) > \left( 1 - \frac{\varepsilon}{2} \right) \frac{m}{n}. \quad (4)$$

On the other hand, by (3),

$$\mathbb{E}[\chi_1] \leq \frac{1}{n} \sum_{u \in V} (d^+(u) + d^\perp(u)) \leq (1 + 2\alpha) \frac{m}{n}, \quad (5)$$

where the last inequality uses Fact 3.5 and  $\sum_{u \in V} d^\perp(u) \leq 2\alpha m$ . Since  $d^+(u) + d^\perp(u) \leq \deg(u)$  for all  $u \in V$ , by (3),

$$\mathbb{E}[\chi_1] \leq \frac{1}{n} \sum_{u \in V} \deg(u) = \frac{2m}{n}. \quad (6)$$

This completes the proof of Claim 3.3 because, using (4),(5) and (6), we get

$$\left( 1 - \frac{\varepsilon}{2} \right) \cdot \frac{m}{n} < \mathbb{E}[\chi_1] \leq \left( 1 + 2 \min \left( \alpha, \frac{1}{2} \right) \right) \cdot \frac{m}{n}. \quad \triangleleft$$

Let random variable  $\chi = \frac{1}{s} \sum_{i=1}^s \chi_i$  denote the mean of  $\chi_i$ 's calculated in Step 9 of Algorithm 3. Since all  $\chi_i$ 's are independent and identically distributed,  $\mathbb{E}[\chi] = \mathbb{E}[\chi_1]$ . Furthermore, the output  $\widehat{d}$  of the algorithm is  $2\chi$  and hence,  $\mathbb{E}[\widehat{d}] = 2\mathbb{E}[\chi]$ . By Claim 3.3, if  $\widehat{d} \geq \bar{d}/8$  then  $\mathbb{E}[\widehat{d}] \leq 2\bar{d}$ . By Markov's inequality,  $\Pr[\widehat{d} > 8\bar{d}] \leq \Pr[\widehat{d} > 4\mathbb{E}[\widehat{d}]] \leq 1/4$ . This completes the proof of part 1 of Lemma 3.2.

Now consider the case when  $\frac{\bar{d}}{8} \leq \widehat{d} \leq 8\bar{d}$ . Observe that  $0 \leq \chi_i \leq 4\sqrt{n\widehat{d}}/\varepsilon$  for all  $i \in [s]$  by Step 6. Hence, by an application of the Hoeffding bound,

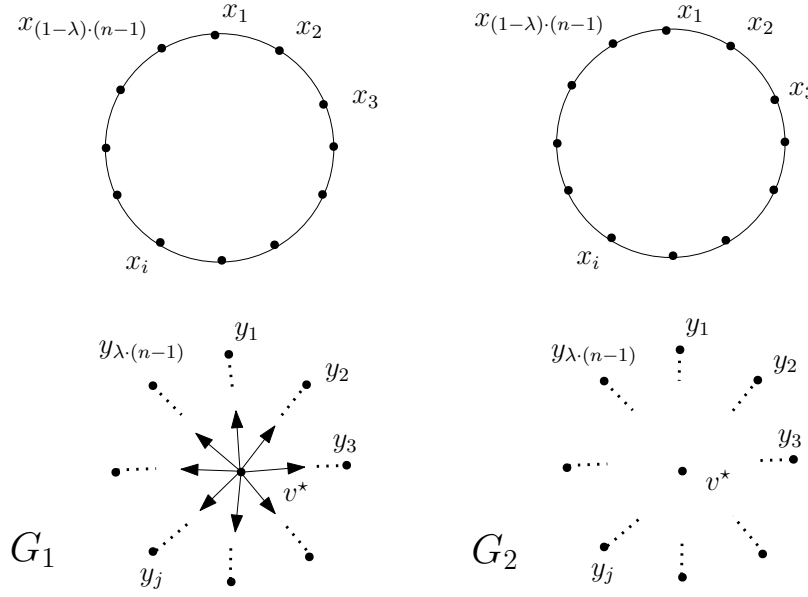
$$\Pr \left[ |\chi - \mathbb{E}[\chi]| \geq \frac{\varepsilon}{2} \cdot \mathbb{E}[\chi] \right] \leq 2 \exp \left( -\frac{\varepsilon^2/4}{2 + \varepsilon/2} \cdot \frac{s \mathbb{E}[\chi]}{4} \sqrt{\frac{\varepsilon}{n\widehat{d}}} \right) < \delta,$$

where we used  $\varepsilon < 1/2$  and  $\widehat{d} \leq 8\bar{d}$  in the simplification. Hence, with probability at least  $1 - \delta$ , we have,  $(1 - \frac{\varepsilon}{2}) \cdot \mathbb{E}[\chi_1] < \chi < (1 + \frac{\varepsilon}{2}) \cdot \mathbb{E}[\chi_1]$ . Since  $\bar{d} = 2\mathbb{E}[\chi]$ , by Claim 3.3, we get that with probability at least  $1 - \delta$ ,

$$\left( 1 - \frac{\varepsilon}{2} \right) \left( 1 - \frac{\varepsilon}{2} \right) \cdot \bar{d} < \widehat{d} < \left( 1 + \frac{\varepsilon}{2} \right) \left( 1 + 2 \min \left( \alpha, \frac{1}{2} \right) \right) \cdot \bar{d},$$

proving part 2 of Lemma 3.2. ◀

The rest of the proof of Theorem 1.7 appears in the full version of this article [20].



■ **Figure 4** The partially erased graphs  $G_1$  and  $G_2$  described in the proof of Theorem 1.8. The dotted lines represent erased entries in the adjacency lists of corresponding vertices. The lines with arrows indicate that the entry corresponds to the vertex to which the arrow points to. The circles represent the  $(1-\lambda)(n-1)$ -cycles.

### 3.2 A Lower Bound for Estimating the Average Degree

In this section, we prove Theorem 1.8.

**Proof of Theorem 1.8.** Fix  $\lambda = \frac{2\alpha}{1+\alpha}$ . Note that  $\lambda \in (0, 1]$  since  $\alpha \in (0, 1]$ . Consider any integer  $n$  such that  $\lambda(n-1)$  is an even integer. Since  $\alpha$  is rational, there are infinitely many such  $n$ . We define two  $n$ -node graphs,  $G_1$  and  $G_2$  (see Figure 4). Both graphs contain a cycle consisting of  $(1-\lambda)(n-1)$  vertices. Of the remaining  $\lambda(n-1) + 1$  vertices, both graphs have  $\lambda(n-1)$  vertices of degree 1, with the only entry in the adjacency list of each such vertex erased. The last vertex, called  $v^*$ , is where  $G_1$  and  $G_2$  differ. In  $G_1$ , we have that  $\text{Adj}(v^*)$  consists of the labels of the  $\lambda(n-1)$  degree-1 vertices. In contrast, in  $G_2$ , the vertex  $v^*$  is isolated.

The graph  $G_1$  can only be completed to a graph consisting of two components: a cycle of length  $(1-\lambda)(n-1)$  and a star consisting of  $\lambda(n-1)$  edges. The graph  $G_2$  can only be completed to a graph consisting of a cycle of length  $(1-\lambda)(n-1)$ , one isolated vertex, and a matching of size  $\lambda(n-1)/2$ . Hence, the total lengths of the adjacency lists of  $G_1$  and  $G_2$  are  $2(n-1)$  and  $(2-\lambda)(n-1)$ , respectively. The number of entries erased in both graphs is  $\lambda(n-1)$ . So, the fraction of erased entries in the adjacency lists of  $G_1$  and  $G_2$  are  $\frac{\lambda}{2}$  and  $\frac{\lambda}{2-\lambda}$ , respectively. Hence, both  $G_1$  and  $G_2$  are  $\alpha$ -erased, as  $\frac{\lambda}{2-\lambda} = \alpha$ . The average degree of  $G_1$  and  $G_2$  are  $\frac{2(n-1)}{n}$  and  $\frac{(2-\lambda)(n-1)}{n}$ , respectively. The ratio of the average degrees is  $\frac{2}{2-\lambda} = 1 + \alpha$ .

The rest of the proof is similar to that of Theorem 1.6. We define two distributions  $\mathcal{D}_1$  and  $\mathcal{D}_2$  as the uniform distributions over the set of all graphs isomorphic to  $G_1$  and  $G_2$ , respectively. To differentiate between the two distributions, any tester must necessarily query  $v^*$  which requires  $\Omega(n)$  queries. The ratio of the average degrees of the two distributions is  $1 + \alpha$ . Hence, to approximate the average degree within a factor of  $(1 + \gamma)$ , where  $\gamma < \alpha$ , any tester must query  $\Omega(n)$  vertices. ◀

## 4 Conclusion and Open Questions

In this work, we initiate the study of sublinear-time algorithms for problems on partially erased graphs. Our investigation opens up a plethora of research directions and possibilities for future work. Next, we discuss several specific open questions arising from our work.

### Phase Transitions in the Complexity of Erasure-Resilient Connectedness Testing

As shown in Section 2, there is a phase transition in the complexity of connectedness testing at  $\alpha = \varepsilon$  from time independent of the size of the graph to  $\Omega(n)$ . Our upper bound on the complexity of this problem exhibits another, less drastic phase transition at  $\alpha = \varepsilon/2$ , when the asymptotic dependence of the running time on  $\varepsilon$  and  $\alpha$  changes. We conjecture that this second phase transition is inherent (and not an artifact of our techniques). It would be interesting to investigate whether connectedness testing when  $\alpha \in [\varepsilon/2, \varepsilon)$  is fundamentally different from the same problem when  $\alpha \in [0, \varepsilon/2)$ .

### Erasure-Resilient Testing of Monotone Properties in the Bounded-Degree Model

A property of a graph is *monotone* if it is preserved under deletion of edges and vertices. That is, if  $G$  satisfies a monotone property then so does every subgraph of  $G$ . Many important graph properties, including bipartiteness, 3-colorability, and triangle-freeness, are monotone.

In the bounded-degree property testing model [16], an  $n$ -node graph  $G$  with the degree bound  $D$  is represented as a concatenation of  $n$  adjacency lists, each of length  $D$ . For a vertex  $v \in G$  and an index  $i \in [D]$ , a neighbor query  $(v, i)$  returns a valid vertex in the graph if  $i \leq \deg(v)$  and a special symbol, say  $\sqcup$ , if  $i > \deg(v)$ . The graph  $G$  is  $\varepsilon$ -far from satisfying a property  $\mathcal{P}$  if at least  $\varepsilon n D$  entries in the adjacency lists of  $G$  need to be modified to make it satisfy  $\mathcal{P}$ .

Bounded-degree property testing can be generalized in a natural way to account for erased entries in adjacency lists. A bounded-degree graph is  $\alpha$ -erased if at most  $\alpha n D$  entries of its adjacency lists are erased. We observe that a tester for a monotone property of bounded-degree graphs can be made erasure-resilient via a simple transformation.

► **Observation 4.1.** *Let  $\mathcal{P}$  be a monotone property of graphs. Suppose there exists an  $\varepsilon$ -tester for  $\mathcal{P}$  in the bounded-degree model that makes  $q(\varepsilon, n, D)$  queries. Then there exists an  $\alpha$ -erasure-resilient  $\varepsilon$ -tester for  $\mathcal{P}$  in the bounded-degree model that makes at most  $D^2 \cdot q(\varepsilon - 2\alpha, n, D)$  queries and works for all  $\alpha \in (0, \varepsilon/2)$ .*

This transformation is not efficient for general graphs, as the maximum degree of a graph can be  $n - 1$ . It is interesting to understand how much erasure-resilience affects query complexity of testing monotone properties in our erasure-resilient model for general graphs.

### Erasure-Resilient vs. Tolerant Testing of Graphs

For  $0 \leq \varepsilon_1 < \varepsilon_2 < 1$ , an  $(\varepsilon_1, \varepsilon_2)$ -tolerant tester for a property  $\mathcal{P}$  must accept, with high probability, if the input is  $\varepsilon_1$ -close<sup>4</sup> to  $\mathcal{P}$  and reject, with high probability, if the input is  $\varepsilon_2$ -far from  $\mathcal{P}$  [24]. Dixit et al. [7] observed that, for properties of functions, erasure-resilient testing is no harder than tolerant testing. Specifically, a tolerant tester for a property of functions can be easily converted to an erasure-resilient tester with the same complexity.

---

<sup>4</sup> An object is  $\varepsilon_1$ -close to a property  $\mathcal{P}$  if it is not  $\varepsilon_1$ -far from  $\mathcal{P}$ .

The new tester can run the tolerant tester, filling in the queried erasures with arbitrary values. However, this argument fails in the case of testing properties of graphs represented as adjacency lists, since the erased entries have to be filled in so that the resulting completion is a valid graph. In the bounded-degree model, we can use a  $(2\alpha, \varepsilon - 2\alpha)$ -tolerant tester for a property  $\mathcal{P}$  to obtain an  $\alpha$ -erasure-resilient  $\varepsilon$ -tester for  $\mathcal{P}$  with an overhead  $O(D^2)$  in query complexity via a transformation similar to the one explained in our discussion of monotone properties. It is an important open question to understand the relationship between erasure-resilient and tolerant testing in the general graph model.

### Symmetric vs. Asymmetric Erasures

Our definition of partially erased graphs is general in the sense that erased entries may be *asymmetric*: an edge  $(u, v)$  can be erased in  $\text{Adj}(u)$ , but not in  $\text{Adj}(v)$ . A partially erased graph has only *symmetric* erasures if it has no half-erased edges, that is,  $u \in \text{Adj}(v)$  iff  $v \in \text{Adj}(u)$  for any two nodes  $u, v$ . It is an interesting direction to investigate which computational tasks are strictly easier in the model with symmetric erasures compared to the model with asymmetric erasures.

---

### References

---

- 1 Maryam Aliakbarpour, Amartya Shankha Biswas, Themis Gouleakis, John Peebles, Ronitt Rubinfeld, and Anak Yodpinyanee. Sublinear-time algorithms for counting star subgraphs via edge sampling. *Algorithmica*, 80(2):668–697, 2018. doi:10.1007/s00453-017-0287-3.
- 2 Sepehr Assadi, Michael Kapralov, and Sanjeev Khanna. A simple sublinear-time algorithm for counting arbitrary subgraphs via edge sampling. In *Proc. of Innovations in Theoretical Computer Science (ITCS)*, pages 6:1–6:20, 2019. doi:10.4230/LIPIcs.ITCS.2019.6.
- 3 Omri Ben-Eliezer, Eldar Fischer, Amit Levi, and Ron D. Rothblum. Hard properties with (very) short PCPPs and their applications. In *Proc. of Innovations in Theoretical Computer Science (ITCS)*, pages 9:1–9:27, 2020. doi:10.4230/LIPIcs.ITCS.2020.9.
- 4 Petra Berenbrink, Bruce Krayenhoff, and Frederik Mallmann-Trenn. Estimating the number of connected components in sublinear time. *Inf. Process. Lett.*, 114(11):639–642, 2014. doi:10.1016/j.ipl.2014.05.008.
- 5 Piotr Berman, Sofya Raskhodnikova, and Grigory Yaroslavtsev.  $L_p$ -testing. In *Proc. of ACM Symposium on Theory of Computing (STOC)*, pages 164–173, 2014. doi:10.1145/2591796.2591887.
- 6 Bernard Chazelle, Ronitt Rubinfeld, and Luca Trevisan. Approximating the minimum spanning tree weight in sublinear time. *SIAM J. Comput.*, 34(6):1370–1379, 2005. doi:10.1137/S0097539702403244.
- 7 Kashyap Dixit, Sofya Raskhodnikova, Abhradeep Thakurta, and Nithin Varma. Erasure-resilient property testing. *SIAM J. Comput.*, 47(2):295–329, 2018. doi:10.1137/16M1075661.
- 8 Talya Eden, Amit Levi, Dana Ron, and C. Seshadhri. Approximately counting triangles in sublinear time. *SIAM J. Comput.*, 46(5):1603–1646, 2017. doi:10.1137/15M1054389.
- 9 Talya Eden, Dana Ron, and C. Seshadhri. Sublinear time estimation of degree distribution moments: The degeneracy connection. In *Proc. of Intl. Colloquium on Automata, Languages and Programming (ICALP)*, pages 7:1–7:13, 2017. doi:10.4230/LIPIcs.ICALP.2017.7.
- 10 Talya Eden, Dana Ron, and C. Seshadhri. Extremely simple algorithm for estimating the number of edges. Personal Communication, 2019.
- 11 Talya Eden, Dana Ron, and C. Seshadhri. On approximating the number of  $k$ -cliques in sublinear time. *SIAM J. Comput.*, 49(4):747–771, 2020. doi:10.1137/18M1176701.
- 12 Talya Eden and Will Rosenbaum. Lower bounds for approximating graph parameters via communication complexity. In *Proc. of Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX-RANDOM)*, pages 11:1–11:18, 2018. doi:10.4230/LIPIcs.APPROX-RANDOM.2018.11.

- 13 Uriel Feige. On sums of independent random variables with unbounded variance and estimating the average degree in a graph. *SIAM J. Comput.*, 35(4):964–984, 2006. doi:10.1137/S0097539704447304.
- 14 Oded Goldreich. *Introduction to Property Testing*. Cambridge University Press, 2017. doi:10.1017/9781108135252.
- 15 Oded Goldreich, Shafi Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *J. ACM*, 45(4):653–750, 1998. doi:10.1145/285055.285060.
- 16 Oded Goldreich and Dana Ron. Property testing in bounded degree graphs. *Algorithmica*, 32(2):302–343, 2002. doi:10.1007/s00453-001-0078-7.
- 17 Oded Goldreich and Dana Ron. Approximating average parameters of graphs. *Random Struct. Algorithms*, 32(4):473–493, 2008. doi:10.1002/rsa.20203.
- 18 Mira Gonen, Dana Ron, and Yuval Shavitt. Counting stars and other small subgraphs in sublinear-time. *SIAM J. Discrete Math.*, 25(3):1365–1411, 2011. doi:10.1137/100783066.
- 19 Tali Kaufman, Michael Krivelevich, and Dana Ron. Tight bounds for testing bipartiteness in general graphs. *SIAM J. Comput.*, 33(6):1441–1483, 2004. doi:10.1137/S0097539703436424.
- 20 Amit Levi, Ramesh Krishnan S. Pallavoor, Sofya Raskhodnikova, and Nithin Varma. Erasure-resilient sublinear-time graph algorithms. *CoRR*, abs/2011.14291, 2020. arXiv:2011.14291.
- 21 Ramesh Krishnan S. Pallavoor, Sofya Raskhodnikova, and Nithin Varma. Improved bounds for  $k$ -connectedness testing. Unpublished manuscript, 2020.
- 22 Ramesh Krishnan S. Pallavoor, Sofya Raskhodnikova, and Erik Waingarten. Approximating the distance to monotonicity of Boolean functions. In *Proc. of ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1995–2009, 2020. doi:10.1137/1.9781611975994.123.
- 23 Michal Parnas and Dana Ron. Testing the diameter of graphs. *Random Struct. Algorithms*, 20(2):165–183, 2002. doi:10.1002/rsa.10013.
- 24 Michal Parnas, Dana Ron, and Ronitt Rubinfeld. Tolerant property testing and distance approximation. *J. Comput. Syst. Sci.*, 72(6):1012–1042, 2006. doi:10.1016/j.jcss.2006.03.002.
- 25 Sofya Raskhodnikova, Noga Ron-Zewi, and Nithin Varma. Erasures vs. errors in local decoding and property testing. In *Proc. of Innovations in Theoretical Computer Science (ITCS)*, pages 63:1–63:21, 2019. doi:10.4230/LIPIcs.ITCS.2019.63.
- 26 Sofya Raskhodnikova and Adam D. Smith. A note on adaptivity in testing properties of bounded degree graphs. *Electronic Colloquium on Computational Complexity (ECCC)*, 13(089), 2006. URL: <http://eccc.hpi-web.de/eccc-reports/2006/TR06-089/index.html>.

# How to Sell Information Optimally: An Algorithmic Study

Yang Cai 

Yale University, New Haven, CT, USA  
yang.cai@yale.edu

Grigoris Velegkas 

Yale University, New Haven, CT, USA  
grigoris.velegkas@yale.edu

---

## Abstract

We investigate the algorithmic problem of selling information to agents who face a decision-making problem under uncertainty. We adopt the model recently proposed by Bergemann et al. [4], in which information is revealed through signaling schemes called *experiments*. In the single-agent setting, any mechanism can be represented as a menu of experiments. Our results show that the computational complexity of designing the revenue-optimal menu depends heavily on the way the model is specified. When all the parameters of the problem are given explicitly, we provide a polynomial time algorithm that computes the revenue-optimal menu. For cases where the model is specified with a succinct implicit description, we show that the tractability of the problem is tightly related to the efficient implementation of a *Best Response Oracle*: when it can be implemented efficiently, we provide an additive FPTAS whose running time is independent of the number of actions. On the other hand, we provide a family of problems, where it is computationally intractable to construct a best response oracle, and we show that it is NP-hard to get even a constant fraction of the optimal revenue. Moreover, we investigate a generalization of the original model by Bergemann et al. [4] that allows multiple agents to compete for useful information. We leverage techniques developed in the study of auction design (see e.g. [5, 1, 6, 7, 8]) to design a polynomial time algorithm that computes the revenue-optimal mechanism for selling information.

**2012 ACM Subject Classification** Theory of computation → Algorithmic game theory

**Keywords and phrases** Mechanism Design, Algorithmic Game Theory, Information Design

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.81

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/2011.14570>.

**Funding** *Yang Cai*: Supported by a Sloan Foundation Research Fellowship and the NSF Award CCF-1942583 (CAREER).

*Grigoris Velegkas*: Partially supported by a PhD Scholarship from the Onassis Foundation and a PhD Scholarship from the Bodossaki Foundation.

**Acknowledgements** We would like to thank Dirk Bergemann for helpful discussions and the anonymous reviewers for their valuable comments and suggestions.

## 1 Introduction

Decision-making heavily relies on information availability. The uneven distribution of information thus enables markets for trading information. Imagine a bank reviewing a loan application. Information about the borrower's financial status clearly influences the bank's lending decisions. In this setting, the bank already has some private knowledge about the borrower, i.e., through prior interactions, but may still be willing to pay to acquire supplemental information to guide its decision-making. Indeed, Equifax, the credit report



© Yang Cai and Grigoris Velegkas;

licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 81; pp. 81:1–81:20

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



agency, provides its business consumers, e.g., banks and credit card companies, a product called Undisclosed Debt Monitoring, that tracks negative information about individual borrowers.<sup>1</sup>

How should the information owner reveal and price the information? We adopt the model introduced by Bergemann et al. [4] to study this problem. Their model involves a data buyer and a data seller. The data buyer faces a decision under uncertainty, and his payoff depends on the *action* he decides to take and the underlying *state* of the world. Initially, the buyer only has some *imperfect* knowledge about the state, i.e. a prior distribution over the possible states. This piece of information is private to the buyer. On the contrary, the data seller knows the state of the world and can sell supplemental information to the buyer. Since the buyer's willingness to pay for supplemental information is determined by the precision of his own prior belief, we refer to the buyer's prior belief as his type. How does the seller optimize her revenue if the buyer's type is assumed to be drawn from a known distribution? To screen heterogeneous buyer types, the seller offers a menu of information products. Each product has the form of a *statistical experiment*, whose result reveals a signal that is correlated with the underlying state. Bergemann et al. [4] investigate what experiments should be included and how to price them. They obtain analytic solution of the revenue-optimal menu in two special cases: (i) the case with only two possible buyer types and (ii) the case with two states and two possible actions for the buyer to choose from. However, general characterization of the revenue-optimal menu remains elusive. In this paper, we initiate the algorithmic study of this problem and investigate the computational complexity for finding the revenue-optimal menu.

Our first result considers an explicit representation of the problem, where the input contains the buyer's type distribution and his payoff for each action and state pair.

**Result 1:** We design an algorithm that computes the revenue-optimal menu in time polynomial in the number of buyer types, the number of buyer actions, and the number of underlying states. (Theorem 3)

For many settings of interest, the model is too expensive to be specified explicitly but has a natural succinct implicit description. Consider the following motivating example. Suppose there is a traffic network  $G$ , and a binary state  $\omega$  that indicates the level of congestion on the edges of  $G$ . A driver wants to go from a vertex  $s$  to another vertex  $t$ , and his payoff is  $H$  minus the expected travel time from  $s$  to  $t$  using the path he picks.<sup>2</sup> Of course, which path is the fastest depends on the driver's belief of the state  $\omega$ . Suppose Waze is offering a service that provides the driver supplemental information about the congestion. How should Waze price its service? In this setting, the driver is the data buyer and the number of actions available to him is exactly the number of  $s - t$  paths, which can be exponential in the size of the network  $G$ . Applying our first result is thus computationally inefficient in this setting. Our second result concerns exactly these settings with succinct implicit descriptions. We show that the key to tractability is the existence of a computationally efficient *Best Response Oracle*, that is, an oracle that accepts a distribution over the underlying state as input and outputs an action with the highest expected payoff. Clearly, in the example above, it is straightforward to construct a computationally efficient best response oracle – simply assign the expected congestion on each edge as its length and run any shortest path algorithm on the graph  $G$ . We show that finding the revenue-optimal menu is tractable as long as there exists a best response oracle.

---

<sup>1</sup> <https://www.equifax.com/business/undisclosed-debt-monitoring/>

<sup>2</sup>  $H$  is a sufficiently large constant so that the payoff is always nonnegative.

**Result II:** For any setting with a constant number of possible underlying states, we design an FPTAS to compute an *up-to- $\varepsilon$  optimal menu*, i.e., a menu whose revenue is at most  $\varepsilon$  less than the optimum, given access to a best response oracle.<sup>3</sup> (Theorem 9)

Without the best response oracle, we show that it is NP-hard to even find a constant factor approximation to the optimal revenue for a family of succinctly describable instances with only two underlying states and one buyer type. (Theorem 12)

We also investigate extensions of the basic model studied in Bergemann et al. [4]. First, we consider the setting where multiple data buyers are competing with each other to receive an informative signal from the data seller. We obtain the following generalization of Result I.

**Result III:** We design an algorithm that computes the revenue-optimal mechanism in time polynomial in the *total number of buyer types*, the number of buyer actions, and the number of underlying states. (Theorem 19)

Note that the straightforward generalization of Result I only gives an algorithm that runs in time polynomial in the *total number of buyer type profiles*, which is exponential in the number of buyers. Our algorithm in Result III runs in time polynomial in the *total number of buyer types*, which is the description size for specifying each buyer's type distribution.

In Section 5, we discuss another natural extension. We can treat a buyer's payoffs for taking actions in various states as the buyer's private information and may be different across buyers. In other words, a buyer type is no longer just the buyer's prior belief of the underlying state but also his payoff function. We show that all our results (c.f. Result I, II, and III) can be easily extended to handle this case, and the modification is summarized in Section 5.

## Our Approach

In the explicit model, we first show that the revenue-optimal menu can be captured by a LP with polynomially many decision variables but exponentially many constraints. We leverage a technique introduced by Chen et al. [15] to transform it to an equivalent LP with polynomially many constraints. In the implicit model with a best response oracle, the main difficulty is that the optimal menu may contain experiments that use an arbitrary number of signals, and as a result the menu cannot even be represented in polynomial time. To overcome this difficulty, we first argue that there always exists an up-to- $\varepsilon$  optimal menu that only contains experiments that use a small number of signals, then apply an algorithm similar to the one in the explicit model to find such a menu. For the multi-agent setting, one can use a LP similar to the one in the explicit model to capture the optimal mechanism. The issue with this approach is that both of the number of variables and constraints are exponential in the number of agents, which is too large to solve. Our solution is inspired by an approach used to computing the revenue-optimal multi-item auctions [5, 1, 6, 7, 8]. The key idea is to first represent mechanisms in a succinct way known as the “reduced forms” and use a LP to search for the revenue-optimal reduced form. However, as a reduced form is only a succinct description, given a reduced form, it is not obvious what mechanism it corresponds to. Indeed, it is not even clear whether it corresponds to any mechanism. The main technical barrier we overcome is to design efficient algorithms to (i) check the feasibility of a reduced form and (ii) to implement a feasible reduced form as an actual mechanism.

---

<sup>3</sup> Our algorithm runs in time polynomial in the total number of buyer types and  $1/\varepsilon$ .

## 1.1 Related Work

### Relationship with Monopoly Pricing

A well-studied problem from mechanism design, the *monopoly pricing problem*, bears some resemblance to our problem. The monopoly pricing problem asks what is the revenue-optimal mechanism to sell one or more items to a buyer, whose valuation/willingness to pay for the items is drawn from a known distribution. In single-dimensional settings, the problem is completely resolved by [35, 36]. In multi-dimensional settings, complete characterizations are known only in several special cases [27, 28, 30, 18, 26, 22, 21], but simple and approximately optimal mechanisms [13, 14, 31, 2, 34, 37, 9, 12], as well as algorithmic characterizations [5, 1, 6, 7, 8] have been discovered in fairly general settings. The nature of information goods adds an extra layer of difficulty to the pricing problem. The value of information is determined by how much such information can improve the quality of decision-making. Buyers with different beliefs do not simply have different values for different experiments, but they may even disagree on their ranking. This richness in buyer valuations does not happen in single-dimensional monopoly pricing, but already exists in single-dimensional information pricing, where there are only two possible underlying states.<sup>4</sup> As a result, the optimal menu in information pricing has a more complex structure. For example, Bergemann et al. [4] showed that even in the two-state case, the seller sometimes needs to use randomized experiments to maximize her revenue. In this work, we show that despite the new challenges of selling information products, some of the algorithmic ideas from monopoly pricing are still useful.

### Relationship with Information Design

Similar to our problem, the designer constructs a signaling scheme that reveals partial information about the underlying state to influence the action of the agents in information design [32]. There has been growing interest in the algorithmic study of information design [24, 23, 25, 16, 19, 10]. The fundamental difference between our setting and information design is that, in our setting, the buyer's action does not have direct effect on the seller's utility. The seller only derives utility from the monetary transfers received from the buyer.

### Relationship with other Information-Selling Models

The work that is most related to the problem we are studying is by Babaioff et al. [3]. Similar to us, they consider a seller who knows the state of the world  $\omega$  and wants to sell information to a buyer whose prior is drawn from some distribution  $\Theta$ . However, there is a subtle but crucial difference between our models. In their work, the seller's information disclosure strategy and the price are allowed to be dependent on the realized state of the world  $\omega$ . By contrast, our model requires the seller to commit to a mechanism before the realization of the state  $\omega$ . Their main results state that (i) when  $\omega$  and the buyer's type are independently distributed, revelation principle holds, i.e., a single-round interaction suffices; and (ii) when  $\omega$  and the buyer's type are correlated, a full surplus extraction mechanism, similar to Crémer and McLean [17], exists and can be computed efficiently. These results differ quite substantially from the structural results for the model we study here [4]. For instance, full surplus extraction is in general impossible in our model. In a recent work, Chen et al. [15] extends the results by Babaioff et al. [3] to the setting where the buyer is budget-constrained, and improves upon some of the algorithmic results.

---

<sup>4</sup> In this case, the buyer's prior belief can be represented using a single real number in  $[0, 1]$ .

## 2 Preliminaries

### Model and Notation

The data buyer faces a decision problem under uncertainty. The state of the world  $\omega$  is drawn from a state space  $\Omega$ . The buyer chooses an action  $a$  from an action space  $A$ . Throughout the paper, we use  $m$  to denote the size of  $A$ . The buyer's **ex-post utility** for choosing action  $a$  under state  $\omega$  is defined to be  $u_{\omega,a}$  and is assumed to lie in  $[0, 1]$ . The buyer has some prior information about the state of the world which is denoted by  $\theta$  and comes from a set  $\Theta \subseteq \Delta\Omega$ . We call  $\theta$  the *type* of the buyer, and use  $\theta_\omega$  to denote the probability that the buyer assigns to the event that the state of the world is  $\omega$ . The type of the buyer is distributed according to  $F$ . Apart from the buyer, there is also a *seller* who observes the state of the world and is willing to sell supplemental information to the buyer. We refer to the buyer as he and to the seller as she.

### Experiment

The seller provides supplemental information to the buyer via a *signaling scheme* which we call *experiment*. A signaling scheme is a commitment to  $|\Omega|$  probability distributions over a set of different signals  $S$ , such that when the state of the world is realized, the seller draws a signal from the corresponding distribution and sends it to the buyer. We denote such an experiment by  $E = (S, \pi(E))$ , where  $\pi(E) : \Omega \rightarrow \Delta S$  denotes the distributions that experiment  $E$  is using. We denote the probability that experiment  $E$  sends signal  $s_k$  when the state of the world is  $\omega$  by  $\pi_{\omega,k}(E) = \Pr[s_k|\omega]$ . In this work, it is useful to think of the experiment  $\pi(E)$  as a matrix whose rows are indexed by the states of the world and columns are indexed by the signals. A *menu* of experiments is a collection  $\mathcal{M} = \{(E, t(E))\}$ , where  $t(E) \in [0, 1]$  is the payment the buyer has to make when he purchases experiment  $E$ . In the multi-agent setting, a menu is insufficient to generate the optimal revenue, and our goal there is to compute the revenue-optimal *mechanism*. In the single-agent setting, the interaction between the seller and the buyer works as follows:

1. The seller posts a menu  $\mathcal{M}$ .
2. The state of the world  $\omega$  and the type of the buyer  $\theta$  are realized.
3. The buyer chooses some experiment  $E$  from the menu based on his type and pays  $t(E)$ .
4. The seller sends the buyer a signal  $s$  that is drawn from  $\pi_{\omega,\cdot}(E)$ .
5. The buyer chooses an action  $a$ , based on his original belief  $\theta$  and the signal  $s$ , and receives utility  $u_{\omega,a}$ .

### The Value of an Experiment

To understand the behavior of the buyer, we first explain how the buyer evaluates an experiment. We first explain how the buyer would act if the only information available to him was his type  $\theta$ . Since the buyer picks an action that maximizes his expected utility, his best move without receiving any additional information from the seller is  $a(\theta) = \arg \max_a \sum_{\omega} \theta_{\omega} u_{\omega,a}$  and his **base utility** following that move is  $u(\theta) = \max_a \sum_{\omega} \theta_{\omega} u_{\omega,a}$ . If he receives extra information from the seller, he updates his beliefs and may choose a new action that induces higher expected value based on his posterior distribution over the states. After receiving signal  $s_k$  from experiment  $E$  his belief about the state of the world is

$$\Pr[\omega|s_k, \theta] = \frac{\theta_{\omega} \pi_{\omega,k}(E)}{\sum_{\omega' \in \Omega} \theta_{\omega'} \pi_{\omega',k}(E)}.$$

Hence, the best action is

$$a(s_k|\theta) \in \arg \max_a \sum_{\omega} \left( \frac{\theta_{\omega} \pi_{\omega,k}(E)}{\sum_{\omega' \in \Omega} \theta_{\omega'} \pi_{\omega',k}(E)} \right) u_{\omega,a},$$

which yields **conditional expected utility**

$$u(s_k|\theta) := \max_a \sum_{\omega} \left( \frac{\theta_{\omega} \pi_{\omega,k}(E)}{\sum_{\omega'} \theta_{\omega'} \pi_{\omega',k}(E)} \right) u_{\omega,a}.$$

When it is clear from the context, we might drop  $\theta$  in the previous expressions. Notice that after computing his posterior, the buyer's conditional expected utility is linear in the actions so we can assume w.l.o.g. that he picks a single action and not a distribution over actions. Taking the expectation over the signal he will receive, we denote the **value of the experiment**  $E$  for type  $\theta$  to be

$$V_{\theta}(E) = \sum_{s_k \in S} \max_a \left\{ \sum_{\omega} \theta_{\omega} \pi_{\omega,k}(E) u_{\omega,a} \right\}.$$

We assume that the buyer is quasilinear. We denote the **expected net utility** of type  $\theta$  for experiment  $E$  as  $V_{\theta}(E) - t(E)$ .

Notice that two different types  $\theta, \theta'$  may have different favorite actions under the same signal. As a result,  $V_{\theta}(E)$  is not a linear function over  $\theta$  for a fixed experiment  $E$  even when the type  $\theta$  is single-dimensional, i.e.,  $|\Omega| = 2$ . This is in sharp contrast to standard single-dimensional auction design settings, where the agent's value is always a linear function over the type when the allocation is fixed.<sup>5</sup>

## IC and IR Menu

The buyer chooses the experiment that gives him the highest net utility, and we slightly abuse notation and denote by  $E(\theta)$  the experiment of the menu  $\mathcal{M}$  that type  $\theta$  prefers

$$E(\theta) = \arg \max_E V_{\theta}(E) - t(E).$$

If all experiments give him expected net utility smaller than his base utility  $u(\theta)$ , he will not purchase any experiment. For convenience, we assume that there is a null experiment that provides no information offered at price 0, so now without loss of generality every type takes an experiment from the menu. We sometimes abuse notation to call a menu *Incentive Compatible* (IC) and *Individually Rational* (IR). By that, we mean that every buyer type  $\theta$  selects the experiment  $(E(\theta), t(\theta))$  that maximizes his expected net utility. Formally, these constraints are captured by the following two sets of inequalities:

$$\begin{aligned} V_{\theta}(E(\theta)) - t(\theta) &\geq V_{\theta}(E(\theta')) - t(\theta'), \quad \forall \theta, \theta' \in \Theta, \\ V_{\theta}(E(\theta)) - t(\theta) &\geq u(\theta), \quad \forall \theta \in \Theta. \end{aligned}$$

Once we have fixed the parameters of the model, we denote by  $\text{REV}(\mathcal{M})$  the revenue that menu  $\mathcal{M}$  generates and by  $\text{OPT}$  the optimal revenue in this setting.

---

<sup>5</sup> Indeed, this linearity even holds for quite general multi-dimensional settings in auction design, for example, when the buyer has additive valuations.

### 3 Optimal Menu for a Single Agent

In this section, we consider the problem of computing the optimal menu for a single agent. As we will show, the computational complexity of the problem is tightly related to the way that the problem is specified. We consider three different models.

- **Explicit model:** the distribution  $F$  and the ex-post utility matrix  $U$  are given explicitly in the input.
- **Implicit model with a best response (BR) oracle:** the distribution  $F$  is given along with a best response oracle. The best response oracle accepts a distribution over the states as input and outputs the action that generates the highest expected utility w.r.t. the input distribution.
- **Implicit model with succinct description:** the distribution  $F$  is given along with the description of a Turing Machine that computes the ex-post utility of any pair of state and action in time polynomial in the description size.

We summarize our results with respect to the three different models: (i) we show that computing the optimal menu in the explicit model is captured by a polynomial size LP; (ii) even though the number of actions in the implicit model with BR oracle may be arbitrarily large, we provide an FPTAS to find an up-to- $\varepsilon$  optimal menu for settings with a constant number of underlying states; (iii) we construct a succinctly representable instance of the problem such that computing a constant factor approximation or an up-to- $\varepsilon$  optimal menu is NP-hard.

#### 3.1 Explicit Model

In this section, we discuss the basic setting where the model is explicitly given. First, we state a structural result of the optimal menu that allows us to restrict the number of signals.

► **Definition 1** (Responsive Experiment). *A buyer type  $\theta$  is responsive to an experiment  $E$  if every signal  $s$  of  $E$  leads  $\theta$  to a different optimal choice of action and, in particular  $a(s_k | \theta) = a_k$  for all  $s_k \in S$ .*

Note that if an experiment  $E$  is responsive to any buyer type,  $E$  has exactly  $m$  signals. An intuitive way to think about Definition 1 is that signal  $s_i$  of  $E$  recommends the buyer to take an action  $a_i$ , and type  $\theta$  is responsive to  $E$  if  $\theta$  always follows the recommendation. Importantly, a different type  $\theta'$  may not be responsive to experiment  $E$  and does not follow the recommendations. We now state the structural result from [4] that states that it is without loss of generality to consider a menu where each buyer type purchases a responsive experiment.

► **Lemma 2** (Adapted from Proposition 1 from [4]). *A menu  $\mathcal{M}$  is responsive if for every buyer type  $\theta$ , it chooses a responsive experiment  $E(\theta)$  from the menu. We define the outcome of a menu as the joint distribution of states, actions, and monetary transfers resulting from every buyer type's optimal choice of experiment and subsequent choice of action. The outcome of every menu can be attained by a responsive menu.*

The proof of Lemma 2 is based on a revelation-principle type of argument. More precisely, assume that some type  $\theta$  prefers an experiment  $E(\theta)$  for which he is not responsive. Then, we can merge all the signals  $s_{i_1}, \dots, s_{i_\ell}$  of  $E(\theta)$  that lead the type to take the same action  $a_k$  and just send the merged signal as the new  $k$ -th signal  $s_k$ . Equipped with the structural result, we are ready to show how to capture the design of the optimal menu as a LP. Since

there exists an optimal menu that is responsive, every experiment in this menu has at most  $m$  signals. For every buyer type  $\theta$ , we use  $\pi(\theta)$  to denote the experiment type  $\theta$  purchases, where  $\pi_{\omega,i}(\theta)$  is the probability to send signal  $s_i$  when the state is  $\omega$ .  $\{\pi_{\omega,i}(\theta)\}_{\omega \in \Omega, i \in [m], \theta \in \Theta}$  is the first set of variables. We also have the prices for each experiment  $\{t(\theta)\}_{\theta \in \Theta}$  as variables. The main difference between selling experiments and selling items is that different types may interpret the same experiment differently. More specifically, for a responsive experiment  $(\pi(\theta), t(\theta))$ , the optimal choice of action for type  $\theta$  after receiving signal  $s_i$  is simply action  $a_i$ ; while for a different type  $\theta'$ , the best action after receiving signal  $s_i$  can be a completely different action  $a_j$  due to a different induced posterior. As our LP is designed to compute the optimal and responsive menu, we need to guarantee that for any type  $\theta$ , purchasing experiment  $(\pi(\theta), t(\theta))$  and following the recommendation is better than purchasing any experiment  $(\pi(\theta'), t(\theta'))$  for a different type  $\theta'$  and subsequently choosing the optimal action based on the induced posterior. We remark that the straightforward formulation of the previous constraints requires one to consider all the possible mappings from signals to actions, resulting in a LP that has exponentially many constraints in  $m$ . Although this huge LP can be solved using the ellipsoid algorithm, we use a technique inspired by Chen et al. [15] that allows us to formulate it using only polynomially many constraints in  $m$ .

See Figure 1 for our LP. Besides the variables representing the experiments and prices, we introduce a new set of variables  $\{z_i(\theta, \theta')\}_{i \in [m], \theta, \theta' \in \Theta}$ .  $z_i(\theta, \theta')$  serves as an upper bound of the conditional expected utility that  $\theta$  gets when he considers misreporting as  $\theta'$  and receives signal  $s_i$ , which is guaranteed by the second set of constraints. The first set of constraints make sure that for any type  $\theta$ , the net utility of purchasing experiment  $E(\theta)$  and following its recommendations is no worse than the utility of purchasing any other experiment and subsequently choosing the best action under each signal. Note that we allow  $\theta'$  to be the same  $\theta$  in the first set of constraints, which guarantees that the menu is responsive. The third set of constraints guarantees that purchasing experiment  $(\pi(\theta), t(\theta))$  is no worse than  $\theta$ 's base utility. We refer to these constraints as the individual rationality (IR) constraints. The last two constraints guarantee that  $\pi(\theta)$  is indeed an experiment.

► **Theorem 3.** *The LP in Figure 1 can be solved in time polynomial in  $m = |A|$ ,  $|\Omega|$ , and  $|\Theta|$ , and its optimal solution is the revenue-optimal menu.*

**Proof of Theorem 3.** The number of constraints and the number of variables of the LP in Figure 1 is polynomial in  $m, |\Omega|, |\Theta|$ , so it is clear that it can be solved in polynomial time. We now argue that its solution is indeed the optimal responsive menu, hence the optimal menu as guaranteed by Lemma 2. Observe that every responsive menu corresponds to a feasible solution of the LP. This is because if  $\{(\pi(\theta), t(\theta))\}_{\theta \in \Theta}$  is a responsive menu, then we can satisfy all the constraints by setting  $z_i(\theta, \theta') = \max_{a_j} \sum_{\omega} \theta_{\omega} \pi_{\omega,i}(\theta') u_{\omega,a_j}$ . Conversely, note that every feasible solution of the LP induces a responsive menu that simply chooses the experiment  $E(\theta)$  as  $(\pi(\theta), t(\theta))$  for each type  $\theta \in \Theta$ . This is because (i) the first and second sets of constraints guarantee that  $E(\theta)$  is a responsive experiment for  $\theta$  (this follows from setting  $\theta'$  to be  $\theta$  in both the first and second set of constraints) and  $V_{\theta}(E(\theta)) - t(\theta) \geq V_{\theta}(E(\theta')) - t(\theta')$ ; (ii) the IR constraints ensure that  $V(E(\theta)) - t(\theta) \geq u(\theta)$ . Hence, the optimal solution corresponds to the revenue-optimal menu. ◀

### 3.2 Implicit Model with a Best Response Oracle

In this section, we study the case where the model is provided implicitly. We prove that, given access to a best response oracle, our algorithm can compute an up-to- $\varepsilon$  optimal menu in time  $\text{poly}\left(|\Theta|, \frac{|\Omega||\Omega|^2}{\varepsilon|\Omega|^2 + |\Omega|}\right)$ , which is independent of the number of actions.



**Variables:**

- $\{\pi_{\omega,i}(\theta)\}_{\omega \in \Omega, i \in [m], \theta \in \Theta}$ , denoting the experiments in the menu.
- $\{t(\theta)\}_{\theta \in \Theta}$ , denoting the prices of the experiments.
- $\{z_i(\theta, \theta')\}_{i \in [m], \theta, \theta' \in \Theta}$ , helper variables.  $z_i(\theta, \theta')$  represents an upper bound of the conditional expected utility of signal  $s_i$  from experiment  $E(\theta')$  for type  $\theta$ .

**Linear Program:**

$$\begin{aligned}
& \max \quad \sum_{\theta \in \Theta} F(\theta) t(\theta) \\
& \text{s.t.} \quad \sum_{i \in [m]} \sum_{\omega \in \Omega} \theta_{\omega} \pi_{\omega,i}(\theta) u_{\omega,a_i} - t(\theta) \geq \sum_{i \in [m]} z_i(\theta, \theta') - t(\theta'), \quad \forall \theta, \theta' \in \Theta \quad (\text{IC}) \\
& \quad \quad z_i(\theta, \theta') \geq \sum_{\omega} \theta_{\omega} \pi_{\omega,i}(\theta') u_{\omega,a_j}, \quad \forall \theta, \theta' \in \Theta, \forall i, j \in [m] \\
& \quad \quad \sum_{i \in [m]} \sum_{\omega \in \Omega} \theta_{\omega} \pi_{\omega,i}(\theta) u_{\omega,a_i} - t(\theta) \geq u(\theta), \quad \forall \theta \in \Theta \quad (\text{IR}) \\
& \quad \quad \sum_{i \in [m]} \pi_{\omega,i}(\theta) = 1, \quad \forall \theta \in \Theta, \omega \in \Omega \\
& \quad \quad \pi_{\omega,i}(\theta) \geq 0, \quad \forall \theta \in \Theta, \forall \omega \in \Omega, \forall i \in [m]
\end{aligned}$$

■ **Figure 1** A linear program to find the revenue-optimal menu in the explicit model.

We first prove a structural result which shows that no matter how large the actual action set  $A$  may be, there is some  $A' \subseteq A$  such that a menu  $\mathcal{M}$  that recommends actions only from  $A'$  has negligible revenue loss compared to  $\text{OPT}$ . Moreover,  $|A'|$  depends only on  $|\Omega|$  and the additive approximation error. We also show that these sets can be computed efficiently.

► **Theorem 4.** *For any constant  $\varepsilon > 0$ , given access to a BR oracle we can compute for each type  $\theta \in \Theta$  a set of actions  $A_{\theta}$  by querying  $O\left(\frac{|\Omega||\Omega|^2}{\varepsilon|\Omega|^2 + |\Omega|}\right)$  times the BR oracle, so that there exists an IC and IR menu  $\mathcal{M}$ , whose experiments all contain no more than  $O\left(\frac{|\Omega||\Omega|^2}{\varepsilon|\Omega|^2 + |\Omega|}\right)$  many signals. Moreover, every type  $\theta$  only uses actions from  $A_{\theta}$  upon receiving any of these signals generated by his experiment. Finally, the revenue of  $\mathcal{M}$  is at least  $\text{OPT} - O(\sqrt{\varepsilon})$ .*

We present some lemmas that are used in the proof of Theorem 4. Firstly, we show that given a menu  $\mathcal{M}$  and  $\varepsilon$ , we can create a menu  $\mathcal{M}'$  with the following two properties: (i) the number of signals that  $\mathcal{M}'$  uses depends only on  $|\Omega|$  and  $\varepsilon$ , and every type  $\theta$  values the new experiment he gets at most  $O(\varepsilon)$  less than his original experiment. The proof is based on the idea that merging signals of an experiment  $E$  that are close does not decrease the value of the experiment by much.

► **Lemma 5.** *Let  $\varepsilon > 0$  be some given constant and let  $E = \{E(\theta)\}_{\theta \in \Theta}$  be a set of experiments, where each  $E(\theta)$  uses an arbitrary number of signals. Then, we can create a set of experiments  $E' = \{E'(\theta)\}_{\theta \in \Theta}$  that uses  $O\left(\left(\frac{|\Omega|}{\varepsilon}\right)^{|\Omega|}\right)$  signals per player such that  $V_{\theta}(E'(\theta)) \geq V_{\theta}(E(\theta)) - 2\varepsilon$ .*

We now argue that when we merge two signals, no type can value any experiment more than he did before.

▷ **Claim 6.** Let  $E(\theta)$  be the initial experiment that is offered to type  $\theta$  and  $E'(\theta)$  the experiment that is offered to  $\theta$  after the merge. Then, for any type  $\theta' \in \Theta$  it holds that  $V_{\theta}(E'(\theta')) \leq V_{\theta}(E(\theta'))$ .

So far we have established the existence of a menu  $\mathcal{M}'$  whose number of signals is significantly smaller than the initial one. However, if we do not have access to  $\mathcal{M}$  we cannot compute  $\mathcal{M}'$ . Lemma 7 shows that we can overcome this issue. By “rounding” the entries of the experiment so that for any type the value this experiment generates does not change much. Now, since there is a small number of signals, and the size of every experiment depends only on  $\varepsilon$  and  $|\Omega|$ , we can do an exhaustive search over the discretized entries.

► **Lemma 7.** *Let  $\{E(\theta)\}_{\theta \in \Theta}$  be a set of experiments, where each  $E(\theta)$  uses signals from  $S$ . We also let  $0 < \delta < 1$  be a given number such that  $1/\delta \in \mathbb{N}$ . Then, we can create a set of experiments  $\{E'(\theta)\}_{\theta \in \Theta}$  that uses signals from  $S'$  with  $|S'| = |S|$ , such that  $E'(\theta)$  is a valid experiment with  $\pi_{\omega, s'}(E'(\theta))$  being a multiple of  $\delta$  for all  $\omega \in \Omega, s' \in S', \theta \in \Theta$ , and  $|V_\theta(E'(\theta')) - V_\theta(E(\theta'))| \leq \delta|S|$ , for all  $\theta, \theta' \in \Theta$ .*

One construction that will be useful in our proofs is the  $\varepsilon$ -IC to IC transformation. Lemma 8 shows that if we have a menu whose IC, IR constraints are violated by at most  $\varepsilon$ , we can modify the prices so that it becomes IC, IR and has negligible  $O(\sqrt{\varepsilon})$  revenue loss. The construction is based on a technique developed [20, 11] and frequently used in the Mechanism Design literature. In the single agent setting, the idea is to offer a small multiplicative discount to all types to make sure that if they want to deviate to some other experiment, this will not be much cheaper than the one they were buying in the initial  $\varepsilon$ -IC menu.

► **Lemma 8.** *Let  $\mathcal{M} = \{E_i, t(E_i)\}_{i \in [k]}$  be a menu with  $k$  experiments. Suppose that the IC, IR constraints are violated by at most  $\varepsilon$ . Then, we can compute a new set of prices  $\{\tilde{t}(E_i)\}_{i \in [k]}$  such that the menu  $\tilde{\mathcal{M}} = \{E_i, \tilde{t}(E_i)\}_{i \in [k]}$  is IR, IC and  $\text{REV}(\tilde{\mathcal{M}}) \geq (1 - \sqrt{\varepsilon})\text{REV}(\mathcal{M}) - \sqrt{\varepsilon} - \varepsilon$ , in time  $O(k)$ .*

We are now ready to present a sketch of the proof for Theorem 4. Assume that we start with the optimal menu  $\mathcal{M}^*$ . By Lemma 5, Claim 6, and Lemma 7, we know that we can modify the experiments in  $\mathcal{M}^*$  so that they use only discretized signals. Moreover, the new menu is approximately IC and IR. We then apply Lemma 8 to obtain a menu that is IC and IR by sacrificing a negligible amount of revenue. Finally, to compute the collection of action sets that the types will choose after receiving the signals, we query BR oracle on all the possible discretized distributions where the signals are drawn from.

So far we have only shown a structural result about the existence of a menu that uses  $O\left(\frac{|\Omega||\Omega|^2}{\varepsilon|\Omega|^2 + |\Omega|}\right)$  signals and gives an additive  $O(\sqrt{\varepsilon})$ -approximation to the optimal revenue. We now argue that we can use a LP (Figure 2) to find such a menu. This is done by combining the result of Theorem 4 and modifying the LP we used in Section 3.1.

► **Theorem 9.** *For any  $\varepsilon > 0$ , any set  $\Theta$  of types, given access to a BR oracle, we can use the LP in Figure 2 to compute a menu  $\mathcal{M}$  that achieves  $\text{REV}(\mathcal{M}) \geq \text{OPT} - O(\sqrt{\varepsilon})$ . The number of queries to the BR oracle is poly  $\left(|\Theta|, \frac{|\Omega||\Omega|^2}{\varepsilon|\Omega|^2 + |\Omega|}\right)$  and the running time is poly  $\left(|\Theta|, \frac{|\Omega||\Omega|^2}{\varepsilon|\Omega|^2 + |\Omega|}\right)$ . Moreover, each experiment contains at most  $O\left(\frac{|\Omega||\Omega|^2}{\varepsilon|\Omega|^2 + |\Omega|}\right)$  many signals, and each type  $\theta$  only chooses actions from a set of actions  $A_\theta$  with size  $O\left(\frac{|\Omega||\Omega|^2}{\varepsilon|\Omega|^2 + |\Omega|}\right)$ .*

So far, the number of experiments in our constructions depends on the number of types  $|\Theta|$ . We show that the number of experiments needed for an up-to- $\varepsilon$  optimal menu is independent of  $|\Theta|$ . We achieve this by dropping experiments that are offered to types who are close in

**Variables:**

- $\{\pi_{\omega,i}(\theta)\}_{\omega \in \Omega, i \in [|A_\theta|], \theta \in \Theta}$ , denoting the experiments in the menu.
- $\{t(\theta)\}_{\theta \in \Theta}$ , denoting the prices of the experiments.
- $\{z_i(\theta, \theta')\}_{i \in [|A_{\theta'}|], \theta, \theta' \in \Theta}$ , helper variables.  $z_i(\theta, \theta')$  represents an upper bound of the conditional expected utility of signal  $s_i$  from experiment  $E(\theta')$  for type  $\theta$ .

**Linear Program:**

$$\begin{aligned}
& \max \quad \sum_{\theta \in \Theta} F(\theta) t(\theta) \\
& \text{s.t.} \quad \sum_{i \in [|A_\theta|]} \sum_{\omega \in \Omega} \theta_{\omega} \pi_{\omega,i}(\theta) u_{\omega, a_{\tau_\theta(i)}} - t(\theta) \geq \sum_{i \in [|A_{\theta'}|]} z_i(\theta, \theta') - t(\theta'), \quad \forall \theta, \theta' \in \Theta \quad (\text{IC}) \\
& \quad z_i(\theta, \theta') \geq \sum_{\omega} \theta_{\omega} \pi_{\omega,i}(\theta') u_{\omega, a_j}, \quad \forall \theta, \theta' \in \Theta, \forall i \in [|A_{\theta'}|], \forall a_j \in A \\
& \quad \sum_{i \in [|A_\theta|]} \sum_{\omega \in \Omega} \theta_{\omega} \pi_{\omega,i}(\theta) u_{\omega, a_{\tau_\theta(i)}} - t(\theta) \geq u(\theta), \quad \forall \theta \in \Theta \quad (\text{IR}) \\
& \quad \sum_{i \in [|A_\theta|]} \pi_{\omega,i}(\theta) = 1, \quad \forall \theta \in \Theta, \omega \in \Omega \\
& \quad \pi_{\omega,i}(\theta) \geq 0, \quad \forall \theta \in \Theta, \forall \omega \in \Omega, \forall i \in [|A_\theta|]
\end{aligned}$$

■ **Figure 2** A linear program to find an approximately revenue-optimal menu in the implicit model.

TV-distance. We show that this leads to a menu that preserves the revenue, is  $O(|\Omega|\varepsilon)$ -IC, IR and has  $O\left(\frac{|\Omega|^{2|\Omega|}}{\varepsilon^{|\Omega|}}\right)$  different experiments. Finally, we apply the  $\varepsilon$ -IC to IC transformation to the modified menu.

► **Lemma 10.** *Let  $\mathcal{M} = \{E(\theta), t(\theta)\}_{\theta \in \Theta}$  be a menu of experiments. Then, for any  $\theta, \theta' \in \Theta$  with  $d_{TV}(\theta, \theta') \leq \varepsilon$  it holds that  $V_\theta(E(\theta')) - t(\theta') \geq V_\theta(E(\theta)) - t(\theta) - 2|\Omega|\varepsilon$ .*

We are now ready to prove that we can create a menu that offers a small number of experiments and loses negligible revenue compared to OPT. We do that in two steps, since we are dealing with an action space and type space that are arbitrary. The first step is to shrink the action space that we are considering. In order to do that, we use Theorem 4 that guarantees the existence of a menu which loses negligible revenue and only considers actions from smaller action spaces. The next step is to divide the state space into regions in which all the types are within  $\varepsilon$  in TV-distance. Lemma 10 shows us that if we consider offering a single experiment to all the types in the same region, their values for the new experiment will not change much compared to the one they were getting. Finally, we apply Lemma 8 to solve the issue that the menu resulting from dropping experiments might not be IC, IR.

► **Theorem 11.** *Consider an environment with a type space  $\Theta$ , action space  $A$  and state space  $\Omega$ . Then, given some  $\varepsilon > 0$  and access to a BR oracle we can find a menu  $\mathcal{M}$  that generates revenue at least  $\text{OPT} - O(\sqrt{\varepsilon})$  and offers at most  $O\left(\frac{|\Omega|^{2|\Omega|}}{\varepsilon^{|\Omega|}}\right)$  experiments, in time  $\text{poly}\left(|\Theta|, \frac{|\Omega|^{|\Omega|^2}}{\varepsilon^{|\Omega|^2+|\Omega|}}\right)$ .*

### 3.3 Implicit Model with Succinct Description

In this section, we consider a setting where the model has a succinct implicit description. We show that no algorithm can obtain even a constant factor approximation to the optimal revenue for this setting, unless  $P = NP$ . To be more precise, we consider the following problem.

**Information Pricing SAT (IP-SAT):** find the revenue-optimal menu in the following setting:

- State space  $\Omega$ , type space  $\Theta$ .
- for each state  $\omega$ ,  $\Phi_\omega$  is a boolean formula in CNF over variables in  $X = \{x_1, \dots, x_n\}$
- Action space  $A$ : all the possible truth assignments of the variables in  $X$
- $u_{\omega,a} = \frac{\# \text{ satisfied clauses of } \Phi_\omega \text{ with assignment } a}{\# \text{ clauses of } \Phi_\omega}$

IP-SAT is a hard problem for the buyer in general, as maximizing his net utility requires solving an NP-hard problem. We show here that designing an approximately revenue-optimal menu for IP-SAT is also computationally intractable for the seller. Of course, it is not even clear what the optimal menu looks like in general for IP-SAT as we only have a limited characterization of the optimal menu. In Theorem 12, we construct a special family of IP-SAT instances with 2 states and 1 buyer type, and show how to reduce SAT to it.

► **Theorem 12.** *For any constant  $\varepsilon > 0$ , there does not exist a polynomial time algorithm  $\mathcal{A}$  that computes an menu with revenue at least  $(1/2 + \varepsilon)OPT - \frac{\varepsilon}{2m+4}$  in the IP-SAT problem with  $m+2$  clauses, unless  $P = NP$ .*

**Proof of Theorem 12.** Let  $\Phi = C_1 \wedge \dots \wedge C_m$  be any SAT instance over variables  $x_1, \dots, x_n$ . We show that given  $\mathcal{A}$  we can decide whether  $\Phi$  is satisfiable. We create the following IP-SAT instance: there are two states  $\omega_1, \omega_2$ , a single type  $\theta = (\frac{1}{2}, \frac{1}{2})$  and we set  $\Phi_{\omega_1} = (C_1 \vee y) \wedge \dots \wedge (C_m \vee y) \wedge (x_1 \vee y) \wedge (\neg x_1 \vee y)$ ,  $\Phi_{\omega_2} = (C_1 \vee \neg y) \wedge \dots \wedge (C_m \vee \neg y) \wedge (x_1 \vee \neg y) \wedge (\neg x_1 \vee \neg y)$  to be the two SAT instances with  $m+2$  clauses that the buyer faces. At each state, the actions are the possible assignments of the variables  $x_1, \dots, x_n, y$ . Let  $\Psi$  be some boolean formula in CNF and  $a$  some assignment of its variables. We define  $z_a(\Psi)$  to be the number of clauses in  $\Psi$  that  $a$  satisfies and  $w(\Psi)$  the total number of clauses in  $\Psi$ . Then, at each state  $\omega$  when the agent chooses assignment  $a$  we define his ex-post utility to be  $u_{\omega,a} = \frac{z_a(\Phi_\omega)}{w(\Phi_\omega)}$ .

From Bergemann et al. [4], we know the optimal menu should contain only the fully informative experiment  $E^*$ , and the price for this experiment is  $V_\theta(E^*) - u(\theta)$ .<sup>6</sup> Clearly,  $V_\theta(E^*) = 1$ , because the buyer can pick  $y$  to be  $T$  and  $F$  in states  $\omega_1, \omega_2$  respectively and satisfy all clauses. We now focus our attention on  $u(\theta)$ . Assume that without receiving any information the buyer decides to set  $y = T$ . This is w.l.o.g. since it is symmetric with the case he decides to set  $y = F$ . When the state is  $\omega_1$ , he satisfies all the clauses. According to his prior, this happens  $1/2$  of the time, so we see that so far  $u(\theta) \geq 1/2$ . Let us consider which assignment he should pick when the state is  $\omega_2$ . Observe that no matter which value he picks for  $x_1$ , he will always satisfy exactly one of the last two clauses in  $\Phi_2$ . Hence, for variables  $x_1, \dots, x_n$  he better pick the assignment  $a$  that maximizes  $z_a(\Phi)$ . Let  $k = \max_a z_a(\Phi)$ . Then  $u(\theta) = 1/2 + (k+1)/(2m+4) = \frac{m+k+3}{2m+4}$ . Hence, the optimal revenue is  $V_\theta(E^*) - u(\theta) = 1 - \frac{m+k+3}{2m+4} = \frac{m-k+1}{2m+4}$ . If  $\Phi$  is satisfiable, then  $k = m$ , so  $OPT = \frac{1}{2m+4}$ . If  $\Phi$  is not satisfiable then  $k \leq m-1$  so  $OPT \geq \frac{2}{2m+4}$ . Now assume that there is such an algorithm  $\mathcal{A}$  and denote  $REV(\mathcal{A})$  the revenue generated by the mechanism output by  $\mathcal{A}$ . If  $\Phi$  is not satisfiable, since  $OPT \geq \frac{2}{2m+4}$ , it must be that  $REV(\mathcal{A}) > \frac{1}{2m+4}$ . On the other hand, if  $\Phi$  is not satisfiable we have that  $REV(\mathcal{A}) \leq \frac{1}{2m+4}$ . Hence, the existence of  $\mathcal{A}$  allows us to distinguish between satisfiable and unsatisfiable SAT formulas. ◀

<sup>6</sup> The fully informative experiment simply sends out a signal to reveal the state.

► **Remark 13.** Since  $m$  is linear in the description length of the problem, given access to a BR oracle, for any  $\varepsilon > 0$  we can set  $\varepsilon' = c(m\varepsilon)^2$ , for some appropriate constant  $c > 0$ , and then apply Theorem 9. Since  $|\Omega| = 2$  and  $|\Theta| = 1$ , the revenue we get is at least  $\text{OPT} - \varepsilon$  and the running time is  $O\left(\frac{1}{(m\varepsilon)^{10}}\right)$ . However, this does not contradict with the result of Theorem 12, since in this setting the BR oracle solves an NP-hard problem.

## 4 Multi-Agent Setting

In this section, we consider a multi-agent generalization of the model by Bergemann et al. [4]. More specifically, we assume that there are  $n$  buyers who are interested in acquiring extra information from the seller and each buyer's ex-post utility only depends on the state of the world and his own action. We further assume that the types of the buyers are drawn independently from their own type distributions. If there is no competition among them, the solution to the problem follows immediately from the single-agent setting, since the seller can offer each agent his optimal menu separately. Thus, we focus on a more interesting case where the buyers are competitors and only *one* of them can receive an informative signal.

### Input Model and New Notation

We first need to introduce some new notation. We use  $\Theta^i$  to denote the type space of buyer  $i$  and  $F^i(\theta^i)$  to denote the probability that buyer  $i$ 's type is  $\theta^i$ . We use  $\Theta$  to denote the set of all type profiles and  $F(\theta)$  to denote  $\times_{i \in [n]} F^i(\theta^i)$ . We assume the action space  $A$  is the same for each buyer  $i$ , but the ex-post utility  $u_{\omega,a}^i$  for choosing action  $a$  under state  $\omega$  may be different for different buyers. We consider the *explicit model*, that is, for each buyer  $i$ , both  $F^i$  and the ex-post utility matrix  $U^i = \{u_{\omega,a}^i\}_{\omega \in \Omega, a \in A}$  are given as input. We use  $u^i(\theta^i)$  to denote the base utility of buyer  $i$  for choosing the best action under distribution  $\theta^i$ .

### Interaction between the Seller and Buyers

The interaction happens in the following order:

1. The seller commits to a mechanism  $\{(\Pi(\theta) = (\Pi^1(\theta), \dots, \Pi^n(\theta))), \{t(\theta) = (t^1(\theta), \dots, t^n(\theta))\}\}_{\theta \in \Theta}$ , and announces the mechanism to all buyers.
2. The types of the buyers  $\theta = (\theta^1, \dots, \theta^n)$  are realized.
3. Each buyer  $i$  **privately** submits his type  $\theta^i$  to the seller.
4. The seller chooses buyer  $i$  as the winner with probability  $p^i(\theta)$ .
5. The seller observes the state of the world  $\omega$  and sends buyer  $i$  a signal  $s$  according to the signaling scheme  $\Pi^i(\theta)$  and charges buyer  $i$  price  $t^i(\theta)$ .
6. Each buyer  $i$  chooses an action  $a^i$  and receives ex-post utility  $u_{\omega,a^i}^i$ .

There are some subtle issues in our model that require further clarification. The most important of them being the following. After the winner has been chosen, does he observe the signaling scheme  $\Pi^i(\theta)$  that the seller uses to generate the signal  $s$ ? In this work, we consider the setting where the signaling scheme  $\Pi^i(\theta)$  is **not revealed** and the winner only observes the realized signal.<sup>7</sup> Some remarks are in order. Firstly, the seller may want to

<sup>7</sup> One may worry that the winner can obtain extra information from the price  $t^i(\theta)$ . To avoid this, we will design a mechanism so that the price for buyer  $i$  only depends on  $i$ 's type  $\theta^i$ . This is without loss of generality, as we can simply set the price to be  $\mathbb{E}_{\theta^{-i}}[t^i(\theta)]$ .

preserve the privacy of the buyers, and revealing  $\Pi^i(\theta)$  allows the winner  $i$  to infer the other buyers' priors. Secondly, hiding the implemented signaling scheme  $\Pi^i(\theta)$  from the winner allows the seller to design a mechanism with less stringent IC constraints and thus generates higher revenue for the seller. This is because the winner does not know the exact experiment he is getting, if he wants to deviate from the recommendation he must map the same signal to the same action for all potential experiments that he may win. Therefore, he would map the signal to an action that induces the highest *expected utility*, where the expectation is over the other bidders' types and the chosen experiment. On the other hand, if the buyer knew which experiment the signal is drawn from, he could use a mapping that is the best for each particular experiment.

Our goal in this section is to design a polynomial time algorithm to find the Bayesian Incentive Compatible (BIC) and Interim Individually Rational (IIR) mechanism that achieves the highest revenue among all BIC and IIR mechanisms for our model. It is not hard to see that Lemma 2 generalizes to our multi-agent setting. We begin by introducing an extension of the LP in Figure 1 to the multi-agent setting. Define  $\Pi^i(\theta) = (\pi_{\omega,j}^i(\theta))_{\omega \in \Omega, j \in [m]}$ ,  $p^i(\theta)$ , and  $t^i(\theta)$  as the decision variables for every buyer  $i$  and type profile  $\theta$ . Recall that  $m = |A|$ .

$$\begin{aligned}
& \max \quad \sum_{\theta \in \Theta} F(\theta) \sum_{i \in [n]} t^i(\theta) \\
& \text{s.t.} \quad \sum_{\theta^{-i}} F^{-i}(\theta^{-i}) \left( \sum_{\substack{\omega \in \Omega, \\ j \in [m]}} \theta_{\omega}^i \pi_{\omega,j}^i(\theta^i, \theta^{-i}) u_{\omega,a_j}^i + (1 - p^i(\theta)) u^i(\theta^i) - t^i(\theta^i, \theta^{-i}) \right) \geq \\
& \quad \sum_{\theta^{-i}} F^{-i}(\theta^{-i}) \left( \sum_{j \in [m]} z_j^i(\theta^i, \tilde{\theta}^i, \theta^{-i}) + (1 - p^i(\tilde{\theta}^i, \theta^{-i})) u^i(\theta^i) - t^i(\tilde{\theta}^i, \theta^{-i}) \right), \quad \forall i, \forall \theta^i, \tilde{\theta}^i \text{ (BIC)} \\
& \quad \sum_{\theta^{-i}} F^{-i}(\theta^{-i}) z_j^i(\theta^i, \tilde{\theta}^i, \theta^{-i}) \geq \sum_{\theta^{-i}} F^{-i}(\theta^{-i}) \sum_{\omega} \theta_{\omega}^i \pi_{\omega,j}^i(\tilde{\theta}^i, \theta^{-i}) u_{\omega,a_k}, \quad \forall i, \forall j, k, \forall \theta^i, \tilde{\theta}^i \\
& \quad \sum_{\theta^{-i}} F^{-i}(\theta^{-i}) \left( \sum_{\substack{\omega \in \Omega, \\ j \in [m]}} \theta_{\omega}^i \pi_{\omega,j}^i(\theta) u_{\omega,a_j}^i + (1 - p^i(\theta)) u^i(\theta^i) - t^i(\theta) \right) \geq u^i(\theta^i), \quad \forall i, \theta^i \text{ (IIR)} \\
& \quad \sum_{j \in [m]} \pi_{\omega,j}^i(\theta) = p^i(\theta), \quad \forall i, \forall \theta, \forall \omega \text{ (feasibility)} \\
& \quad \sum_{i \in [n]} p^i(\theta) \leq 1, \quad \forall \theta \text{ (feasibility)} \\
& \quad \pi_{\omega,j}^i(\theta) \geq 0, \quad \forall i, \forall \theta, \forall \omega, \forall j \text{ (feasibility)}
\end{aligned}$$

Observe that the number of variables is exponential in  $n$  and the number of constraints is exponential in both  $n$  and  $m$ . There is no hope to solve this LP in polynomial time. The main challenge is how to remove the exponential dependence on  $n$ . To overcome this obstacle, we use a method that is powerful in the study of multi-item auctions, that is, rewriting the LP using a more succinct representation of the mechanism known as the reduced form [5, 1, 6, 7, 8]. We first define the reduced form of a mechanism.

► **Definition 14 (Reduced Form).** *Given a mechanism  $\mathcal{M} = (\{\Pi(\theta)\}_{\theta \in \Theta}, \{t(\theta)\}_{\theta \in \Theta})$ , we define its reduced form  $\{\hat{\Pi}^i(\theta^i)\}_{i \in [n], \theta^i \in \Theta^i}$ , where  $\hat{\Pi}^i(\theta^i) = \{\hat{\pi}_{\omega,j}^i(\theta^i)\}_{\omega \in \Omega, j \in [m]}$  and  $\hat{\pi}_{\omega,j}^i(\theta^i) = \mathbb{E}_{\theta^{-i}}[\pi_{\omega,j}^i(\theta)]$  for each state  $\omega$  and  $j \in [m]$ , and its interim prices  $\{\hat{t}^i(\theta^i)\}_{i \in [n], \theta^i \in \Theta^i}$ , where  $\hat{t}^i(\theta^i) = \mathbb{E}_{\theta^{-i}}[t^i(\theta)]$ . We use  $\mathcal{P}(F)$  to denote the set of all reduced forms for a particular type distribution  $F$ .*

It is not hard to see that  $\mathcal{P}(F)$  is a closed convex set, as the set of all mechanisms is clearly closed and convex, and  $\mathcal{P}(F)$  is simply a linear transformation of that set. Intuitively, the reduced form is the “expected experiment and price” that the each buyer believes he will be allocated when his type is realized, and the expectation is taken over the randomness of the other buyers’ types.

► **Lemma 15.** *For any type distribution  $F$ ,  $\mathcal{P}(F)$  is a closed convex set.*

The LP in Figure 3 searches for the reduced form of the revenue-optimal mechanism. Notice that the size of the reduced-form LP is substantially smaller than the original one, and the number of variables is polynomial in the number of agents. With these new variables we can still express the BIC and IIR constraints of the initial LP. However, it is not yet clear how to check whether these variables correspond to an actual feasible mechanism. In the next section, we show how to design a separation oracle that checks the feasibility efficiently.

#### Variables:

- $\{\hat{\pi}_{\omega,j}^i(\theta^i)\}_{\omega \in \Omega, i \in [n], \theta^i \in \Theta^i, j \in [m]}$ , denoting the reduced form of the mechanism.
- $\{\hat{t}^i(\theta^i)\}_{i \in [n], \theta^i \in \Theta^i}$ , denoting the interim prices.
- $\{\hat{p}^i(\theta^i)\}_{i \in [n], \theta^i \in \Theta^i}$ , denoting the allocation probabilities of the experiment
- $\{\hat{z}_j^i(\theta^i, \tilde{\theta}^i)\}_{i \in [n], j \in [m], \theta^i, \tilde{\theta}^i \in \Theta^i}$ , helper variables.  $\hat{z}_j^i(\theta^i, \tilde{\theta}^i)$  represents an upper bound of the conditional expected utility of signal  $s_j$  for type  $\theta^i$ .

#### Linear Program:

$$\begin{aligned}
 \max \quad & \sum_{i \in [n]} \sum_{\theta^i \in \Theta^i} F^i(\theta^i) \hat{t}^i(\theta^i) \\
 \text{subject to} \quad & \sum_{j \in [m]} \sum_{\omega \in \Omega} \theta_{\omega}^i \hat{\pi}_{\omega,j}^i(\theta^i) u_{\omega,a_j}^i + (1 - \hat{p}^i(\theta^i)) u^i(\theta^i) - \hat{t}^i(\theta^i) \geq \\
 & \sum_{j \in [m]} \hat{z}_j^i(\theta^i, \tilde{\theta}^i) + (1 - \hat{p}^i(\tilde{\theta}^i)) u^i(\theta^i) - \hat{t}^i(\tilde{\theta}^i), \quad \forall i, \forall \theta^i, \tilde{\theta}^i \quad (\text{BIC}) \\
 & \hat{z}_j^i(\theta^i, \tilde{\theta}^i) \geq \sum_{\omega \in \Omega} \theta_{\omega}^i \hat{\pi}_{\omega,j}^i(\tilde{\theta}^i) u_{\omega,a_k}^i \quad \forall i, \forall \theta^i, \tilde{\theta}^i, \forall j, k \\
 & \sum_{j \in [m]} \sum_{\omega \in \Omega} \theta_{\omega}^i \hat{\pi}_{\omega,j}^i(\theta^i) u_{\omega,a_j}^i + (1 - \hat{p}^i(\theta^i)) u^i(\theta^i) - \hat{t}^i(\theta^i) \geq u^i(\theta^i), \quad \forall i, \forall \theta^i \quad (\text{IIR}) \\
 & \hat{p}^i(\theta^i) = \sum_{j \in [m]} \hat{\pi}_{\omega,j}^i(\theta^i) \quad \forall i, \forall \omega, \forall \theta^i \\
 & \{\hat{\pi}_{\omega,j}^i(\theta^i)\}_{\omega \in \Omega, i \in [n], \theta^i \in \Theta^i, j \in [m]} \in \mathcal{P}(F) \quad (\text{Feasibility})
 \end{aligned}$$

■ **Figure 3** A linear program to find the reduced form of the revenue-optimal mechanism in the multi-agent setting.

## 4.1 Feasibility of Reduced Forms

To design a separation oracle for the set  $\mathcal{P}(F)$ , we invoke the equivalence between Optimization and Separation in Linear Programming [29, 33], which states that being able to optimize any linear function over a convex set  $P$  is equivalent to having a separation oracle for  $P$ . It is a well-known fact that given a separation oracle for  $P$  one can optimize any linear function using the ellipsoid method. Interestingly, the reverse is also true. If there is an algorithm to optimize any linear function over  $P$ , one can construct a separation oracle for  $P$  using the ellipsoid method. We state a strengthened version of the equivalence due to Cai et al. [7].



The reason that we need to be able to decompose a feasible point into corners of the polytope is that we eventually need to be able to implement the reduced forms as a feasible mechanism. We elaborate more on this later.

► **Theorem 16** (Adapted from Theorem H.1 of [7]). *Let  $P$  be a  $d$ -dimensional closed convex region, and let  $\mathcal{A}$  be any polynomial-time algorithm that takes any direction  $w \in \mathbb{R}^d$  as input and outputs the extreme point  $\mathcal{A}(w) \in P$  in direction  $w$  such that  $\mathcal{A}(w) \cdot w \geq \max_{x \in P} x \cdot w$ . Then we can design a polynomial time separation oracle  $SO$  for  $P$  such that, whenever  $SO(x) = \text{“yes”}$ , the execution of  $SO$  explicitly finds directions  $w_1, \dots, w_k$  such that  $x$  lies in the convex hull of  $\{\mathcal{A}(w_1), \dots, \mathcal{A}(w_k)\}$ .*

To apply this equivalence, we need to show how we can optimize a linear function over the set of feasible reduced-form variables. Recall that we use  $\Pi^i(\theta) = (\pi_{\omega,j}^i(\theta))_{\omega \in \Omega, j \in [m]}$  and  $\hat{\Pi}^i(\theta^i) = (\hat{\pi}_{\omega,j}^i(\theta^i))_{\omega \in \Omega, j \in [m]}$  to denote the ex-post signaling scheme and its reduced form. We will treat  $\Pi^i(\theta)$  and  $\hat{\Pi}^i(\theta^i)$  as  $m|\Omega|$ -dimensional vectors. The following maximization problem plays a crucial role in our approach.

► **Definition 17.** *Consider any type profile  $\theta$ . Let  $\{X^i(\theta^i)\}_{i \in [n], \theta^i \in \Theta^i}$  be a collection of  $m|\Omega|$ -dimensional vectors. We define a Virtual Payoff Maximizer (VPM) w.r.t. these weight vectors  $VPM(\{X^i(\theta^i)\}_{i \in [n], \theta^i \in \Theta^i})$  to be the ex-post signaling scheme  $\Pi(\theta)$  that maximizes the following quantity  $\sum_i \Pi^i(\theta^i) \cdot X^i(\theta^i)$  for every type profile  $\theta$ . The corresponding reduced form  $\hat{\Pi}(\theta)$  is called  $rVPM(\{X^i(\theta^i)\}_{i \in [n], \theta^i \in \Theta^i})$ . In order to ensure that the maximizer is unique, we break ties lexicographically.*

When there is no confusion, we also write  $VPM(w), rVPM(w)$  as the maximizers for the weight vector  $w$ .

► **Lemma 18.** *Given an arbitrary collection of weights  $\{X^i(\theta^i)\}_{i \in [n], \theta^i \in \Theta^i}$  we can find the exact optimal solution of*

$$\max_{\hat{\Pi} \in \mathcal{P}(F)} \sum_{i \in [n]} \sum_{\theta^i \in \Theta^i} \hat{\Pi}^i(\theta^i) \cdot X^i(\theta^i) \text{ in time } O\left(m|\Omega| \left(\sum_{i \in [n]} |\Theta^i|\right) + \left(\sum_{i \in [n]} |\Theta^i|\right)^2\right).$$

**Proof of Lemma 18.** We first rewrite the maximization problem

$$\begin{aligned} \max_{\hat{\Pi} \in \mathcal{P}(F)} \sum_{i \in [n]} \sum_{\theta^i \in \Theta^i} \hat{\Pi}^i(\theta^i) \cdot X^i(\theta^i) &= \max_{\Pi} \sum_{i, \theta^i, \theta^{-i}} F^{-i}(\theta^{-i}) \Pi^i(\theta^i, \theta^{-i}) \cdot X^i(\theta^i) = \\ &= \max_{\Pi} \sum_{i, \theta} F(\theta) \Pi^i(\theta) \cdot \frac{X^i(\theta^i)}{F^i(\theta^i)} = \max_{\Pi} \sum_{\theta} F(\theta) \sum_i \Pi^i(\theta) \cdot \tilde{X}^i(\theta^i), \end{aligned}$$

where  $\tilde{X}^i(\theta^i) = \frac{X^i(\theta^i)}{F^i(\theta^i)}$ .

Let  $\theta$  be a type profile. We now characterize the solution of  $\max_{\Pi(\theta)} \sum_i \Pi^i(\theta) \cdot \tilde{X}^i(\theta^i)$ . If we allocate the experiment to buyer  $i$ , the maximum value we can derive is  $v^i(\theta^i) = \sum_{\omega} \max_a \tilde{X}_{\omega,a}^i(\theta^i)$ . Clearly, the optimal solution of the linear function above is to always allocate the experiment to the buyer with the largest  $v^i(\theta^i)$ .

The ex-post signaling scheme that maximizes  $\sum_{\theta} F(\theta) \sum_i \Pi^i(\theta) \cdot \tilde{X}^i(\theta^i)$  is the one that always allocates the experiment to the buyer with the largest  $v^i(\theta^i)$  for every type profile  $\theta$ . To solve  $\max_{\hat{\Pi} \in \mathcal{P}(F)} \sum_{i \in [n]} \sum_{\theta^i \in \Theta^i} \hat{\Pi}^i(\theta^i) \cdot X^i(\theta^i)$ , we only need to calculate the reduced form of this ex-post signaling scheme and we denote it using  $\hat{\Pi}_*$ .

We first compute  $v^i(\theta^i) = \sum_{\omega} \max_a x_{\omega,a}^i(\theta^i)$  for every buyer  $i$  and every type  $\theta^i$ . This step takes time  $O\left(m|\Omega| \left(\sum_{i \in [n]} |\Theta^i|\right)\right)$ , and there are  $\sum_{i \in [n]} |\Theta^i|$  different such values. Next, for each buyer, we sort  $v^i(\theta^i)$ . This step takes time  $O\left(\sum_{i \in [n]} |\Theta^i| \log |\Theta^i|\right)$ . To compute  $\hat{\Pi}_*^i(\theta^i)$ , we only need to calculate the probability of the event that over the random draws of  $\theta_{-i}$ , there exists another buyer  $\ell \neq i$  either  $v_{\ell}(\theta^{\ell}) > v^i(\theta^i)$  or  $\ell < i$  and  $v_{\ell}(\theta^{\ell}) = v^i(\theta^i)$ . This probability can be computed in time  $O\left(\sum_{i \in [n]} |\Theta^i|\right)$  for each buyer  $i$  and type  $\theta^i$ . Hence, in total we can optimize the linear function in time  $O\left(m|\Omega| \left(\sum_{i \in [n]} |\Theta^i|\right) + \left(\sum_{i \in [n]} |\Theta^i|\right)^2\right)$ . ◀

Combining Theorem 16 and Lemma 18, we have a polynomial time algorithm to solve the LP in Figure 3, but we still need to turn the reduced form into an ex-post signaling scheme. We again use an idea from computing the optimal multi-item auctions, that is, first decomposing the optimal reduced form into a distribution over extreme points of  $\mathcal{P}(F)$ , then implementing all the extreme points that appear in the distribution using a VPM ex-post signaling scheme.

► **Theorem 19.** *We design an algorithm to compute the revenue-optimal mechanism in time  $\text{poly}\left(n, m, |\Omega|, \sum_{i \in [n]} |\Theta^i|\right)$ . Moreover, the mechanism can be implemented as a distribution over  $m|\Omega| \left(\sum_{i \in [n]} |\Theta^i|\right) + 1$  VPM ex-post signaling schemes.*

## 5 Further Extensions and Future Directions

In this section, we discuss further generalizations of the model by Bergemann et al. [4] and future research directions that we believe are interesting to pursue.

### 5.1 Extensions of the Original Model

#### Enlarged Buyer Type

Recall that in the original model the only private information of the buyer is his private belief of the underlying state, which is realized at the beginning of the interaction with the seller. Importantly, the payoffs are public knowledge and remain the same across different buyers. A natural generalization one can consider is to allow the buyer to draw not only his prior belief  $\theta$ , but also his payoff function  $u : \Omega \times A \rightarrow [0, 1]$  from some distribution.

To be more specific, we consider the setting where a buyer's type  $\rho = (\theta, u)$  is drawn from some distribution  $F$  at the beginning of the interaction between the buyer and the seller, and  $\rho$  is private to the buyer. As in the original model, we assume that the seller has access to this distribution. We remark that all of our positive results from Section 3 and Section 4, except for Theorem 11, hold in this extended model as well. The only difference in our constructions is that instead of indexing the variables by  $\theta$  we now index them by  $\rho$ .

#### Misspecified Model

Another generalization we consider in the single-agent setting is the *misspecified* model. In this model the seller has access to some type distribution  $\tilde{F}$  which is within  $\varepsilon$  in TV-distance with the real type distribution  $F$ . Moreover, the seller has access to a type space  $\tilde{\Theta}$  with the following two properties:  $|\tilde{\Theta}| = |\Theta|$ , for all  $\tilde{\theta} \in \tilde{\Theta}$  there is some  $\theta \in \Theta$  for which  $d_{TV}(\tilde{\theta}, \theta) \leq \varepsilon$ . Then, the menu that the seller designs for the misspecified distributions can be modified so that it guarantees only a negligible revenue loss when it is evaluated in the true setting. Lemma 20 formalizes this claim.

► **Lemma 20.** *Let  $\tilde{F}, F$  be the distributions of the types that the seller has access to and the true distribution of the types, respectively. Let also  $\{\tilde{\theta}_i\}_{i \in [k]}, \{\theta_i\}_{i \in [k]}$  be the types that the seller has access to and the true types, respectively. Assume that  $d_{TV}(\tilde{F}, F) \leq \varepsilon_1, d_{TV}(\tilde{\theta}_i, \theta_i) \leq \varepsilon_2, \forall i \in [k]$ . We also let  $\tilde{\mathcal{M}} = \{\tilde{E}(\tilde{\theta}_i), \tilde{t}(\tilde{\theta}_i)\}_{i \in [k]}$  be an IC, IR menu that has revenue  $REV(\tilde{\mathcal{M}})$  under the misspecified distributions and uses at most  $|S|$  signals. Then, we can compute a set of prices  $\{t(\tilde{\theta}_i)\}_{i \in [k]}$  so that  $\mathcal{M} = \{\tilde{E}(\theta_i), t(\theta_i)\}_{i \in [k]}$  is IR, IC and has  $REV(\mathcal{M}) \geq REV(\tilde{\mathcal{M}}) - O\left(\varepsilon_1 + \sqrt{|\Omega|\varepsilon_2}\right)$  under the true distributions.*

Note that Lemma 20 allows us to generalize our results and obtain approximately-optimal menus when we only have black-box access to the distribution of the types. That is, we take enough samples to learn the distribution within Total Variation distance  $\varepsilon$  and then apply Lemma 20.

## 5.2 Future Directions

We believe that the design of Information Markets is a very important problem that has not received sufficient attention by the Theory of Computation community. There are many interesting questions waiting to be addressed.

1. In the single-buyer setting where we only have access to the action space via a BR oracle the running time of our algorithms is exponential in the number of states. An immediate question to ask is whether we can get an FPTAS or even a PTAS that has a better dependence on the number of states.
2. In the multi-agent setting, we consider the case in which the seller does not reveal the signaling scheme that she uses to send a signal to the winner. An interesting question is whether we can have efficient algorithms in the setting where the seller reveals the signaling scheme to the buyer.
3. Currently, in the multi-agent setting we assume that the ex-post utility of each buyer depends only on the state of the world and the action he takes. Is the problem of designing the optimal mechanism when the ex-post utilities also depend on the actions of the other buyers tractable?

---

## References

- 1 Saeed Alaei, Hu Fu, Nima Haghpanah, Jason Hartline, and Azarakhsh Malekian. Bayesian Optimal Auctions via Multi- to Single-agent Reduction. In *the 13th ACM Conference on Electronic Commerce (EC)*, 2012.
- 2 Moshe Babaioff, Nicole Immorlica, Brendan Lucier, and S. Matthew Weinberg. A Simple and Approximately Optimal Mechanism for an Additive Buyer. In *the 55th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2014.
- 3 Moshe Babaioff, Robert Kleinberg, and Renato Paes Leme. Optimal mechanisms for selling information. In *Proceedings of the 13th ACM Conference on Electronic Commerce*, pages 92–109, 2012.
- 4 Dirk Bergemann, Alessandro Bonatti, and Alex Smolin. The design and price of information. *American economic review*, 108(1):1–48, 2018.
- 5 Yang Cai, Constantinos Daskalakis, and S. Matthew Weinberg. An Algorithmic Characterization of Multi-Dimensional Mechanisms. In *the 44th Annual ACM Symposium on Theory of Computing (STOC)*, 2012.
- 6 Yang Cai, Constantinos Daskalakis, and S. Matthew Weinberg. Optimal Multi-Dimensional Mechanism Design: Reducing Revenue to Welfare Maximization. In *the 53rd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2012.

- 7 Yang Cai, Constantinos Daskalakis, and S. Matthew Weinberg. Reducing Revenue to Welfare Maximization : Approximation Algorithms and other Generalizations. In *the 24th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2013.
- 8 Yang Cai, Constantinos Daskalakis, and S. Matthew Weinberg. Understanding Incentives: Mechanism Design becomes Algorithm Design. In *the 54th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2013.
- 9 Yang Cai, Nikhil R. Devanur, and S. Matthew Weinberg. A duality based unified approach to bayesian mechanism design. In *the 48th Annual ACM Symposium on Theory of Computing (STOC)*, 2016.
- 10 Yang Cai, Federico Echenique, Hu Fu, Katrina Ligett, Adam Wierman, and Juba Ziani. Third-party data providers ruin simple mechanisms. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 4(1):1–31, 2020.
- 11 Yang Cai, Argyris Oikonomou, Grigoris Velezgas, and Mingfei Zhao. An efficient epsilon-bic to bic transformation and its application to black-box reduction in revenue maximization. In *the 32nd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2021. [arXiv:1911.10172](#).
- 12 Yang Cai and Mingfei Zhao. Simple mechanisms for subadditive buyers via duality. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 170–183, 2017. [doi:10.1145/3055399.3055465](#).
- 13 Shuchi Chawla, Jason D. Hartline, and Robert D. Kleinberg. Algorithmic Pricing via Virtual Valuations. In *the 8th ACM Conference on Electronic Commerce (EC)*, 2007.
- 14 Shuchi Chawla, Jason D. Hartline, David L. Malec, and Balasubramanian Sivan. Multi-Parameter Mechanism Design and Sequential Posted Pricing. In *the 42nd ACM Symposium on Theory of Computing (STOC)*, 2010.
- 15 Yiling Chen, Haifeng Xu, and Shuran Zheng. Selling information through consulting. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2412–2431. SIAM, 2020.
- 16 Yu Cheng, Ho Yee Cheung, Shaddin Dughmi, Ehsan Emamjomeh-Zadeh, Li Han, and Shang-Hua Teng. Mixture selection, mechanism design, and signaling. In *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, pages 1426–1445. IEEE, 2015.
- 17 Jacques Cremer and Richard P. McLean. Full extraction of the surplus in bayesian and dominant strategy auctions. *Econometrica*, 56(6):1247–1257, 1988.
- 18 Constantinos Daskalakis, Alan Deckelbaum, and Christos Tzamos. Strong duality for a multiple-good monopolist. *Econometrica*, 85(3):735–767, 2017.
- 19 Constantinos Daskalakis, Christos Papadimitriou, and Christos Tzamos. Does information revelation improve revenue? In *Proceedings of the 2016 ACM Conference on Economics and Computation*, pages 233–250. ACM, 2016.
- 20 Constantinos Daskalakis and S. Matthew Weinberg. Symmetries and Optimal Multi-Dimensional Mechanism Design. In *the 13th ACM Conference on Electronic Commerce (EC)*, 2012.
- 21 Nikhil Devanur, Kira Goldner, Raghuvansh Saxena, Ariel Schwartzman, and S Matthew Weinberg. Optimal mechanism design for single-minded agents. *arXiv preprint*, 2020. [arXiv:2002.06329](#).
- 22 Nikhil R Devanur and S Matthew Weinberg. The optimal mechanism for selling to a budget constrained buyer: The general case. In *Proceedings of the 2017 ACM Conference on Economics and Computation*, pages 39–40, 2017.
- 23 Shaddin Dughmi. On the hardness of signaling. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 354–363. IEEE, 2014.
- 24 Shaddin Dughmi, Nicole Immorlica, and Aaron Roth. Constrained signaling in auction design. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 1341–1357. Society for Industrial and Applied Mathematics, 2014.

- 25 Shaddin Dughmi and Haifeng Xu. Algorithmic bayesian persuasion. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 412–425. ACM, 2016.
- 26 Amos Fiat, Kira Goldner, Anna R Karlin, and Elias Koutsoupias. The fedex problem. In *Proceedings of the 2016 ACM Conference on Economics and Computation*, pages 21–22, 2016.
- 27 Yiannis Giannakopoulos and Elias Koutsoupias. Duality and optimality of auctions for uniform distributions. In *ACM Conference on Economics and Computation, EC '14, Stanford , CA, USA, June 8-12, 2014*, pages 259–276, 2014. doi:10.1145/2600057.2602883.
- 28 Yiannis Giannakopoulos and Elias Koutsoupias. Selling two goods optimally. In *Automata, Languages, and Programming - 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part II*, pages 650–662, 2015. doi:10.1007/978-3-662-47666-6\_52.
- 29 Martin Grötschel, László Lovász, and Alexander Schrijver. The Ellipsoid Method and its Consequences in Combinatorial Optimization. *Combinatorica*, 1(2):169–197, 1981.
- 30 Nima Haghpanah and Jason D. Hartline. Reverse mechanism design. *CoRR*, abs/1404.1341, 2014. arXiv:1404.1341.
- 31 Sergiu Hart and Noam Nisan. Approximate Revenue Maximization with Multiple Items. In *the 13th ACM Conference on Electronic Commerce (EC)*, 2012.
- 32 Emir Kamenica and Matthew Gentzkow. Bayesian persuasion. *American Economic Review*, 101(6):2590–2615, 2011.
- 33 Richard M. Karp and Christos H. Papadimitriou. On linear characterizations of combinatorial optimization problems. *SIAM J. Comput.*, 11(4):620–632, 1982. doi:10.1137/0211053.
- 34 Xinye Li and Andrew Chi-Chih Yao. On revenue maximization for selling multiple independently distributed items. *Proceedings of the National Academy of Sciences*, 110(28):11232–11237, 2013.
- 35 Roger B. Myerson. Optimal Auction Design. *Mathematics of Operations Research*, 6(1):58–73, 1981.
- 36 John Riley and Richard Zeckhauser. Optimal selling strategies: When to haggle, when to hold firm. *The Quarterly Journal of Economics*, 98(2):267–289, 1983.
- 37 Andrew Chi-Chih Yao. An n-to-1 bidder reduction for multi-item auctions and its applications. In *SODA*, 2015. arXiv:1406.3278.

# Computation over the Noisy Broadcast Channel with Malicious Parties

**Klim Efremenko**

Ben Gurion University of the Negev, Beer Sheva, Israel  
klimefrem@gmail.com

**Gillat Kol**

Princeton University, NJ, USA  
gillat.kol@gmail.com

**Dmitry Paramonov**

Princeton University, NJ, USA  
dp20@cs.princeton.edu

**Raghuvansh R. Saxena**

Princeton University, NJ, USA  
rrsaxena@princeton.edu

---

## Abstract

We study the  $n$ -party *noisy broadcast channel* with a constant fraction of *malicious parties*. Specifically, we assume that each non-malicious party holds an input bit, and communicates with the others in order to learn the input bits of all non-malicious parties. In each communication round, one of the parties broadcasts a bit to all other parties, and the bit received by each party is flipped with a fixed constant probability (independently for each recipient). How many rounds are needed?

Assuming there are no malicious parties, Gallager gave an  $\mathcal{O}(n \log \log n)$ -round protocol for the above problem, which was later shown to be optimal. This protocol, however, inherently breaks down in the presence of malicious parties.

We present a novel  $n \cdot \tilde{\mathcal{O}}(\sqrt{\log n})$ -round protocol, that solves this problem even when almost half of the parties are malicious. Our protocol uses a new type of error correcting code, which we call a *locality sensitive code* and which may be of independent interest. Roughly speaking, these codes map “close” messages to “close” codewords, while messages that are not close are mapped to codewords that are very far apart.

We view our result as a first step towards a theory of *property preserving interactive coding*, i.e., interactive codes that preserve useful properties of the protocol being encoded. In our case, the naive protocol over the noiseless broadcast channel, where all the parties broadcast their input bit and output all the bits received, works even in the presence of malicious parties. Our simulation of this protocol, unlike Gallager’s, preserves this property of the original protocol.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Communication complexity

**Keywords and phrases** Broadcast Network, Malicious Parties, Communication Complexity

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.82

**Related Version** <https://eccc.weizmann.ac.il/report/2021/001/>

**Funding** *Klim Efremenko*: Supported by the Israel Science Foundation (ISF) through grant No. 1456/18 and European Research Council Grant number: 949707.

*Gillat Kol*: Supported by an Alfred P. Sloan Fellowship, the National Science Foundation CAREER award CCF-1750443, and by the E. Lawrence Keyes Jr. / Emerson Electric Co. Award.

*Dmitry Paramonov*: Supported by the National Science Foundation CAREER award CCF-1750443.

*Raghuvansh R. Saxena*: Supported by the National Science Foundation CAREER award CCF-1750443.

**Acknowledgements** We thank Huacheng Yu for helpful discussions.



© Klim Efremenko, Gillat Kol, Dmitry Paramonov, and Raghuvansh R. Saxena;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 82; pp. 82:1–82:19



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

## 1 Introduction

The field of *interactive coding*, pioneered by Schulman [16, 17, 18], asks the following question:

*Let  $\Pi$  be a communication protocol designed to work over a noiseless channel. Can  $\Pi$  be converted to a noise resilient protocol  $\Pi'$  with similar communication complexity?*

Many works, mainly over the last decade, give affirmative answers to this question for the two-party channel, as well as various multi-party distributed channels. For example, it was shown that protocols in the extensively studied message passing model (peer-to-peer channels) and in the shared blackboard model (broadcast channel), can be *simulated* by protocols that tolerate stochastic noise, i.e., noise that flips each of the communicated bits with constant probability [15, 1, 4, 7, 12, 6, 13, 5]. These only incur a small (sub-logarithmic and in many cases, even constant) multiplicative overhead to the communication. Here, by “simulate” we mean that the new protocols retain the same input-output behavior as the original protocols<sup>1</sup>.

**Property preserving interactive coding.** While the simulation protocols  $\Pi'$ , designed by the aforementioned interactive coding works, are communication efficient and preserve the input-output behavior of the original protocols  $\Pi$ , they often lose the “structure” of the original protocols together with some of the basic properties making the original protocols useful. For instance, the importance of celebrated distributed protocols for the *consensus* and the *leader election* problems stems from their fault tolerance properties – the fact that they keep the same input-output behavior, even in the presence of *malicious parties* that may exhibit crashes or even Byzantine failures<sup>2</sup>.

We study the above interactive coding question in a different light: Assume that the original communication protocol  $\Pi$  satisfies a special property  $P$ , can  $\Pi$  be converted to a noise resilient protocol  $\Pi'$  that still satisfies  $P$ ? Specifically, we focus on protocols with the property  $P = “\Pi$  is resilient to a constant fraction of malicious parties”, and give a simulation protocol  $\Pi'$  over a noisy channel that also satisfies  $P$ .

**The noisy broadcast channel.** In this paper, we consider this new “property preserving” interactive coding question in the *noisy broadcast model*, a noisy version of the shared blackboard model, first suggested by El Gamal in 1984 [8]. In this model, a set of  $n$  parties, each holding a private input, communicate over a noisy broadcast channel. In each round, one of the parties broadcasts a bit to all the other parties, and the bit received by each of the other parties is flipped with some constant probability  $\epsilon > 0$  (independently for each recipient).

We revisit the basic problem suggested by El Gamal, regarding the computation of the *identity function* over the noisy broadcast channel: assume that each party receives a single bit as an input and that the parties’ mutual goal is for all parties to learn all input bits. That is, party  $i$  gets a bit  $x^i \in \{0, 1\}$  and needs to output  $f(x^1, x^2, \dots, x^n) = (x^1, x^2, \dots, x^n)$ . How many communication rounds are needed? Observe that, over the noiseless broadcast

<sup>1</sup> The parties participating in the distributed protocol are assumed to each have an input at the beginning of the protocol and give an output when the protocol terminates. We often think of the entire transcript received by a party as its output.

<sup>2</sup> Recall, for example, that in the consensus problem, each party gets an input bit and all parties need to output the input bit of one of the parties (and, in particular, all parties need to output the same bit). Indeed, there are short trivial protocols with the required input-output behavior, but these are not resilient to malicious parties.



channel, this can be done in  $n$  rounds by simply having each party broadcast its bit. In 1988, Gallager [7] showed that  $\mathcal{O}(n \log \log n)$  broadcast rounds suffice to solve the problem over the noisy broadcast channel, with polynomially small error probability. Note that this also means that any function on  $n$  input bits can be computed over the noisy broadcast channel in  $\mathcal{O}(n \log \log n)$  rounds<sup>3</sup>, making the identity function “complete” for the computation of such functions. Gallager’s result was shown to be tight in the beautiful 2005 paper by Goyal, Kindler, and Saks [10].

**The noisy broadcast channel with malicious parties.** We consider El Gamal’s question in the presence of malicious parties. Specifically, assume that a constant fraction of the parties participating in the protocol are malicious. These parties are controlled by a know-it-all adversary that sees all inputs, as well as all the sent and received bits. The rest of the parties are assumed to be honest and following the protocol, and they may not know the identity of the malicious parties.

Due to the presence of malicious parties, it is unrealistic to expect all parties to compute the identity function (neither over the noiseless broadcast channel, nor over the noisy broadcast channel) for two reasons: (i) the malicious parties will deliberately give incorrect outputs; (ii) a malicious party with an input  $x^i$  can behave as if it is holding the input  $1 - x^i$ , preventing the honest parties from learning its input. Therefore, we relax our requirement and only ask that each *honest* party outputs the input bits of all other *honest* parties. Specifically, we want that for every input  $x$ , the following condition is satisfied with high probability:

(\*) *Each party  $i$  outputs  $(\tilde{x}_1^i, \tilde{x}_2^i, \dots, \tilde{x}_n^i)$ , where if both  $i$  and  $i'$  are honest, then  $\tilde{x}_{i'}^i = x^{i'}$  (otherwise,  $\tilde{x}_{i'}^i$  can be arbitrary).*

Over the noiseless broadcast channel, it is easy to see that the simple aforementioned  $n$ -round protocol, where each party broadcasts its bit once and outputs all the bits it received, satisfies this relaxed condition. How many rounds of communication are needed when we work over the noisy broadcast channel?

## 1.1 Our Results

The main result of this paper is a novel  $n \cdot \tilde{\mathcal{O}}(\sqrt{\log n})$ -round protocol for computing the identity function under the relaxed condition (\*), in the presence of a constant fraction of malicious parties and stochastic noise. While more costly than Gallager’s protocol, which utterly breaks down in the presence of even a single malicious party (see discussion in Subsection 2.3), our protocol does beat the naive  $\mathcal{O}(n \log n)$  protocol<sup>4</sup>.

We note that our protocol assumes the *statistical* variant of the noisy broadcast channel (see, e.g., [12, 13]), where the noise flips every sent bit with probability exactly  $\epsilon$ . Generalizing the protocol for the *fault tolerant* noisy broadcast channel where the noise can flip the  $j^{\text{th}}$  bit received by the  $i^{\text{th}}$  party with a different probability for different  $i$ s and  $j$ s, as long as these probabilities are all between 0 and  $\epsilon$ , is left open (see more about this in Subsection 2.6).

<sup>3</sup> To compute the function  $g(x^1, x^2, \dots, x^n)$ , the parties run the protocol that computes the identity function. After the protocol, each party knows  $x^1, x^2, \dots, x^n$  and can evaluate  $g(\cdot)$  by itself.

<sup>4</sup> In the naive protocol, each party broadcasts its bit  $\Theta(\log n)$  times. Party  $i$  outputs  $(\tilde{x}_1^i, \tilde{x}_2^i, \dots, \tilde{x}_n^i)$ , where  $\tilde{x}_{i'}^i$  is the majority of the bits party  $i$  received from party  $i'$ . If both parties  $i$  and  $i'$  are honest, the bit  $\tilde{x}_{i'}^i$  is the input of party  $i'$ , except with polynomially small probability, and by a union bound, our relaxed property holds, except with polynomially small probability.

► **Theorem 1.** *Let  $\epsilon, \theta < 1/2$  and  $n$  be large enough. There exists an  $n \cdot \tilde{O}(\sqrt{\log n})$ -round randomized protocol with private and public randomness<sup>5</sup> over the noisy broadcast channel with noise rate exactly  $\epsilon$ , that computes the identity function in the presence of a  $\theta$ -fraction of malicious parties (i.e., satisfies condition  $(*)$ ) with error  $1/n$ . Furthermore, the protocol is computationally efficient – the algorithm for each party runs in almost-linear, i.e.,  $n^{1+o(1)}$ , time.*

As discussed above, due to the “completeness” of the identity function, our result also implies that *any*  $n$ -bit function (each party gets a single input bit) can be computed over the noisy broadcast channel in the presence of a constant fraction of malicious parties in  $n \cdot \tilde{O}(\sqrt{\log n})$  rounds.

## 1.2 Our Techniques and the Notion of Locality Sensitive Codes

The starting point of our construction is Gallager’s protocol. At the heart of this protocol is a clever trick that uses (standard) error correcting codes. While Gallager’s trick inherently breaks in the presence of malicious parties, we draw inspiration from his ideas and design a different protocol using a new type of codes that we call *locality sensitive codes*.

Roughly speaking, our locality sensitive codes map “close” messages to “close” codewords, while messages that are not close are mapped to codewords that are very far apart. In more detail, our alphabet set is the set of integers. Two messages  $m, m' \in \mathbb{Z}^k$  that are close in every coordinate ( $|m_i - m'_i| \leq \alpha$  for every  $i \in [k]$ ) will be mapped to codewords that are close in almost every coordinate, but two messages that are far apart in at least one coordinate are mapped to codewords that are far in almost all coordinates. Note that locality sensitive codes are a generalization of classical error correcting codes. Indeed, by setting  $\alpha = 0$  we can interpret “closeness” as “equality” (two messages are close only if they are identical) and retrieve the definition of standard error correcting codes – identical messages are mapped to identical codewords and non-identical messages are mapped to codewords with a large distance.

We mention that the definition of locality sensitive codes is reminiscent of that of *locality sensitive hash functions*, often used by algorithms for the *approximate nearest neighbor* problem and other related problems (see, e.g., [2, 3]). However, while locality sensitive hash functions are hash functions, and as such, are contracting the message, locality sensitive codes stretch the message. Since the problem studied in this paper is, at least seemingly, very different from other known applications of locality sensitive hash functions, devising further connections between locality sensitive codes and locality sensitive hashes will be interesting.

## 1.3 Future Directions

Our work suggests several future directions, we list a few below.

**Improving our result.** One obvious interesting question is whether our result in Theorem 1 is optimal in terms of communication complexity, or whether it can be improved. In particular, is it possible to match Gallager’s construction and design an  $\mathcal{O}(n \log \log n)$  protocol for the

---

<sup>5</sup> Observe that private and public randomness are “incomparable” in our model: the public random string is known to all parties, including the malicious parties. The private random strings are each known to a single party, and, in particular, the private random string of an honest party is not known to any of the malicious parties (see Subsection 3.3).

identity function that handles stochastic noise and also works in the presence of a constant fraction of malicious parties<sup>6</sup>? One direction towards this goal is to improve the construction of locality sensitive error correcting codes.

**Interactive codes preserving other properties.** In this paper, we consider the property  $P = \text{“}\Pi \text{ is resilient to a constant fraction of malicious parties”}$  and show that in certain cases, protocols  $\Pi$  that satisfy  $P$  can be compiled into noise resilient protocols that still respect  $P$ . Can this be done for other useful properties  $P$ ? We mention that work of [9] can be interpreted as asking a similar question with  $P = \text{“}\Pi \text{ is computing some function } f \text{ privately”}$  (and with adversarial noise), and answering it in the negative.

**General interactive coding with malicious parties.** As mentioned in Subsection 1.1, Theorem 1 implies a similar simulation for any  $n$ -bit function. Thus, one could have hoped that our simulation would also extend to functions with more than  $n$  input bits (i.e., the case where parties get long inputs), allowing us to convert any  $t$ -round noiseless protocol with malicious parties to a  $t \cdot \tilde{O}(\sqrt{\log n})$ -round protocol over the noisy broadcast channel with malicious parties. However, even without the presence of malicious parties, we suspect that the overhead in making a broadcast protocol noise resilient can be a multiplicative factor of  $\tilde{\Omega}(\log n)$  (and, in particular, a multiplicative  $\mathcal{O}(\log \log n)$  overhead à la Gallager does not always suffice).

**Interactive coding with malicious parties over different models.** While we believe that there is no scheme with a small overhead that converts protocols in the noiseless broadcast channel that are resilient to malicious parties to protocols in the noisy broadcast channel that are resilient to malicious parties, such a scheme may exist for other channels, such as the (synchronous or asynchronous) peer-to-peer model. If it does, this scheme would imply noise resilient consensus and leader election protocols in the respective models. We mention that noise resilient consensus is considered in [11], under a different noise model.

**Better than optimal interactive coding with adversarial noise.** Another interesting direction is further exploring the relaxed requirement (\*). In this paper, we design a protocol for the identity function that satisfies (\*) even in the presence of a constant fraction of malicious parties and stochastic noise. Does such a protocol exist in the presence of a more general type of noise – say, if we allow the adversary to corrupt a different set of parties in each round or if we allow general adversarial noise, where the adversary can corrupt a constant fraction of the received bits? While prior works argue that in multi-party settings it is impossible to handle more than a  $1/n$  fraction of adversarial noise, as with this budget, the adversary can corrupt one of the parties completely, this may be possible under a relaxed definition along the lines of (\*).

## 2 Overview of Our Protocol

In this section, we build up to our protocol step by step, covering our main ideas.

### 2.1 The Identity Problem Over the Broadcast Channel

In the broadcast channel, there are  $n$  parties that communicate with one another. This communication happens through bit “broadcasts”, namely, bits sent from one of the  $n$  parties to *all* the parties. The party sending these bits computes them using the bits it

<sup>6</sup> Goyal et al. [10], proved their lower bounds for the statistical model assumed in this paper, where each received bit is flipped with probability exactly  $\epsilon$ . Therefore, our result cannot be improved to an  $o(n \log \log n)$ -round protocol.

received during the communication that happened so far and its own private input. The end-goal of this communication is to compute a joint function of all the private inputs while communicating as few bits as possible.

An important setting (indeed, complete in certain respects) is when all the  $n$  parties have a bit as their input, and they want to know the inputs of all the other parties. Formally, party  $i \in [n]$  has a bit  $x^i$  and should output the bit string  $x^1, x^2, \dots, x^n$  after the communication. We shall call this problem the identity problem in the rest of this document.

The identity problem admits a simple and optimal  $n$  round communication protocol over the broadcast channel: In round  $i \in [n]$ , party  $i$  broadcasts their bit  $x^i$  to all the parties. After  $n$  rounds, all the parties would have received all the bits and can output the string  $x^1, x^2, \dots, x^n$ .

This simple protocol boasts of some nice and non-trivial properties. For example, even if an (arbitrarily large) subset of parties do not follow the protocol and are malicious, we still have the guarantee that all non-malicious parties will output the bit of all other non-malicious parties correctly. This property makes sure that, when this protocol is run on a large distributed system, it will be resilient to a subset of the parties failing or being taken over by an adversary. Moreover, the protocol described is also communicationally and computationally efficient.

However, this protocol also has a major weakness in that it crucially relies on the fact that the channel does not corrupt any of the bits sent, which are received exactly by all the parties. Is it possible to have a protocol that is resilient to both malicious parties and channel corruptions?

## 2.2 The Noisy Broadcast Channel and Gallager's Protocol

To study the above question, one needs to move to the noisy broadcast channel. This channel is identical to the broadcast channel except that it has stochastic noise, namely, there is a parameter  $0 < \epsilon < \frac{1}{2}$  such that when any of the parties broadcasts a bit  $b$  over the channel, all the parties may either receive  $b$ , with (independent) probability  $1 - \epsilon$ , or may receive the bit  $1 - b$ , with probability  $\epsilon$ .

The protocol for the identity problem described above can even be simulated over the noisy broadcast channel, albeit with higher communication. For example, one may repeat each bit broadcast during the protocol  $\mathcal{O}(\log n)$  times, and this will ensure that all the parties receive the bit sent except with probability polynomially small in  $n$ . A simple union bound over all the  $n$  parties and all the  $n$  bits shows that the protocol does solve the identity problem except with probability polynomially small in  $n$ .

In fact, not only does this protocol solve the identity problem in a way that is resilient to channel corruptions, it also preserves the property of the original protocol of being resilient to malicious parties. Indeed, even if an arbitrarily large subset of parties in the protocol are malicious, all the non-malicious parties will output the bits of all the malicious parties with high probability.

### 2.2.1 Gallager's Protocol [7]

The main drawback of this protocol is that it communicates  $\mathcal{O}(n \log n)$  bits and it is not immediately clear if this is optimal in terms of communication. In fact, without the restriction of being resilient to malicious parties, Gallager, in his elegant work [7], showed that it is provably not so, exhibiting a protocol that communicates  $\mathcal{O}(n \log \log n)$  bits and is resilient to channel corruptions<sup>7</sup>.

<sup>7</sup> [10] later showed that Gallager's protocol is optimal up to constant factors.

The main idea behind Gallager’s improved protocol can be easily understood from an information theoretic viewpoint. Consider the  $\mathcal{O}(\log n)$  rounds where a given party broadcasts in the simple  $\mathcal{O}(n \log n)$  communication protocol. In the first few rounds when this party broadcasts, each bit broadcast gives a relatively large amount of information about its input to all the other parties. However, as the number of repetitions increases, the other parties already know the input bit with significant probability and each bit sent starts to give lesser and lesser information about the party’s input. Thus, by independently repeating their inputs later in the protocol, the parties are wasting a lot of communication to convey a small amount of information about their inputs. This is clearly suboptimal and a more sensible approach is to “combine” the inputs of several of the parties into fewer bits, while maintaining that the information conveyed is the same.

Actually implementing this idea requires ingenuity, and [7] does it by having a protocol with three stages as described below<sup>8</sup>:

- **Stage Broadcast:** This stage of Gallager’s protocol has  $\mathcal{O}(n \log \log n)$  rounds and in this stage, all the parties broadcast their input  $\mathcal{O}(\log \log n)$  times. As the total number of broadcasts per party is small (only  $\mathcal{O}(\log \log n)$ ), all the broadcasts in this stage convey a large amount of information to the other parties.

Because all the parties broadcast  $\mathcal{O}(\log \log n)$  times in stage **Broadcast**, after this stage, any party can decode (by simple majority based decoding) the input of any other party correctly, except with probability at most  $\frac{1}{\text{polylog}(n)}$ .

- **Stage Guess:** In this stage, the parties divide themselves into families of size  $A = \Theta(\log n)$ . As the size of the families is  $\Theta(\log n)$ , a simple union bound shows that any party in a family can decode the inputs of all other parties in the family correctly, except with probability at most  $\frac{1}{\text{polylog}(n)}$ .

In fact, as the noise received by all the parties is independent, we also have concentration and, except with probability polynomially small in  $n$ , we have that at least 90% of the parties in a family correctly decode the inputs of all the parties in the family.

- **Stage Boost:** This is the most important stage of the protocol, where the parties “combine” multiple input bits and give information about all of the combined bits in the same communication bit. Recall that before this stage, the parties are divided into families of size  $A$ , and, except with probability polynomially small in  $n$ , at least 90% of the parties in a family know the input of all the parties in the family.

We describe stage **Boost** from the perspective of a single family, noting that the behavior of all the families is symmetric. Party  $i$  in this family, for all  $i \in [A]$ , takes the vector of bits it decoded for all the parties in the family, encodes this vector using a constant rate error correcting code, and broadcasts the  $i^{\text{th}}$  coordinate of this encoding<sup>9</sup>. Observe that as this coordinate depends on all the decoded bits, transmitting it conveys information about all the  $A$  bits in the family. Indeed, it is this combination that reduces the failure probability (“boosts” the success probability) of the protocol from  $\frac{1}{\text{polylog}(n)}$  to  $\frac{1}{\text{poly}(n)}$  without wasting  $\Omega(\log n)$  communication per party and allows us to union bound over all parties and all bits.

Due to the fact that 90% of the parties in a family encode the same (correct) vector of inputs, at least 90% of the sent coordinates are actually coordinates from one codeword. As the channel corrupts each sent symbol with a small constant probability, for any one of the  $n$  parties, it holds, except with probability polynomially small in  $n$ , that at least

<sup>8</sup> We note that [7] does not describe his protocol in terms of these stages. However, it will be helpful for us to talk about his protocol in this framework.

<sup>9</sup> We assume in this description that the error correcting code takes a bit string to a string over a larger alphabet but of the same length, and assume for simplicity that the parties can broadcast symbols from this larger alphabet in one round.

80% of the received coordinates come from the same (correct) codeword. Because this is a codeword of a good error correcting code, these 80% of the coordinates suffice to decode the inputs of all parties in the family correctly, except with probability polynomially small in  $n$ , and a union bound over all families and all parties finishes the proof of correctness of Gallager's protocol.

## 2.3 Gallager's Protocol in the Presence of Malicious Parties

Gallager's protocol shows that if resilience to malicious parties is not required, then the simple  $\mathcal{O}(n \log n)$  bit communication protocol can be improved. Our main result is stronger, showing that it is possible to do better than the  $\mathcal{O}(n \log n)$  bit communication protocol while maintaining its resilience against malicious parties. Before we describe our ideas, however, it will be helpful to first understand where Gallager's protocol fails if some of the parties are malicious.

At first glance, one may mistakenly believe that Gallager's protocol is also resilient to malicious parties. Indeed, the bits broadcast by the parties in stage **Broadcast** do not depend on the bits received by them, no communication happens in stage **Guess**, and the argument described in Subsubsection 2.2.1 would work (with slightly different parameters) as long as no family has a lot, say more than 10%, of malicious parties. As the families are size  $A = \Theta(\log n)$ , this last property can be ensured, except with probability polynomially small in  $n$ , by, say, partitioning the parties into families randomly before the execution of the protocol.

However, delving deeper reveals a major problem. Recall that, for stage **Boost** to work, stage **Guess** must ensure that, except with probability polynomially small in  $n$ , at least 90% of the parties in any family have the same (correct) decoding for the input bits of all the parties in the family. As the parties perform majority based decoding in stage **Guess**, this is equivalent to saying that, for at least 90% of the parties in a family, the majority bit they receive for all the parties in the family in stage **Broadcast** is the same (and correct) except with probability polynomially small in  $n$ .

The last property is true if there are no malicious parties. Indeed, all parties repeatedly broadcast their input in stage **Broadcast**, and because the channel only corrupts with a small constant probability, a simple concentration argument shows that, except with probability polynomially small in  $n$ , the majority bit received by at least 90% of the parties in a family will be the same as the bit sent.

However, if one of the parties in a family is malicious, it may, in stage **Broadcast**, broadcast the bit 0 half the time and the bit 1 the other half of the time. This implies that the majority bit received by any other party is equally likely to be 0 or 1, and only around 50% of the parties in the family can agree on the inputs of all the other parties in the family. To make matters worse, if a constant fraction (and not just 1) of the parties in the family are malicious, and all of them behave this way, then no two parties in the family will agree on the inputs of all the parties in the family (with large probability).

Before describing how our protocol gets around this problem, we quickly note that the source of this problem is *not* that the parties perform majority based decoding and decode to a bit only if it is heard at least 50% of the time. Even if the parties have another threshold, say 70%, the malicious parties can simply broadcast 0 in 30% of their broadcasts in stage **Broadcast**, and 1 in the remaining 70%, and cause the same problem.

## 2.4 Our Approach: Boosting Before Guessing

We define the “true count” of a party to be the number of times it broadcasts 1 in stage **Broadcast**. The example from the foregoing section shows that, when there is a malicious party  $i$  whose true count is half of the total broadcasts by party  $i$  in stage **Broadcast**, then majority based decoding implies that no more than 50% of the parties in the family of party  $i$  agree on the bit they decoded for party  $i$ .

Nonetheless, the fact that the channel corrupts only a small constant fraction of the bits sent implies that, except with probability at most  $\frac{1}{\text{polylog}(n)}$ , all the parties in the family of party  $i$  can *approximately* decode the true count for party  $i$ . Our main idea is to run stage **Boost** on these true counts directly, without first executing stage **Guess** to decode the true counts to bits.

Indeed, if we can run stage **Boost** on the true counts, and get all the parties to agree on an approximation for the true counts that is correct except with probability polynomially small in  $n$ , we can union bound over all parties and all counts to conclude that, except with probability polynomially small in  $n$ , all the parties know all the true counts approximately correctly. Once this happens, the parties can discard all counts that are close to  $\frac{1}{2}$ , as these can only be from malicious parties, and run stage **Guess** on the remaining counts to get the input bits of all the non-malicious parties correctly.

### 2.4.1 Computing the Encodings

There is however, a key difference between boosting the true counts and boosting the guessed bits. While boosting the guessed bits, Gallager’s protocol ensured that most of the parties in a family have the same Boolean vector  $u \in \{0, 1\}^A$  of the bits guessed for parties in the family. This allowed an error correcting code of  $u$  to be computed in a distributed way, where every party  $i$  in the family contributes coordinate  $i$  by encoding its Boolean vector.

With the true counts, this is no longer possible as *no* party in the family is likely to have the vector of true counts exactly, and the parties merely possess a good approximation of it. That is, if  $p \in \mathbb{Z}^A$  denotes the vector of the true counts for the family, each party in the family now has a different approximation  $q \in \mathbb{Z}^A$  of  $p$ . The task to be solved now is to compute an encoding of  $p$  in a *distributed* manner, when parties only have access to the approximations  $q$ !

This task seems to be impossible for standard error correcting codes, for which changing any of the coordinates of the message being encoded even slightly may result in a codeword completely unrelated to the original codeword. What we need instead is an error correcting code that is also “locality sensitive”, namely, for two messages (not necessarily Boolean)  $q_1$  and  $q_2$  that are close in all the coordinates, the encodings of  $q_1$  and  $q_2$  are also close in most of the coordinates. On the other hand, if  $q_1$  and  $q_2$  are far in any of the coordinates, then their encodings must be far in almost all of the coordinates.

**Connection to locality sensitive hashing.** Our description above may remind the reader of the area of locality sensitive hashing, where vectors in  $\mathbb{R}^d$  are hashed into buckets such that vectors that are close to each other are likely to be hashed into the same bucket, while vectors that are far apart are likely to be hashed to different buckets.

Our description of locality sensitive error correcting codes is related, but different. Firstly, by definition, a hash function loses information to make the data more manageable while an error correcting code adds more redundant information to make the data resilient to corruptions. Moreover, a locality sensitive hash only requires the hash values of two inputs that are far to be different, while in our definition of locality sensitive error correcting codes,



we will require the codewords to not only be different, but also far apart, in most of the coordinates. In fact, how far they can be will be critical in determining the communication complexity of our protocol.

## 2.5 Locality Sensitive Error Correcting Codes

As described above, we desire a code  $C$  that is locality sensitive. Namely, it has the following properties:

1. (“close”  $\rightarrow$  “close”): If two messages  $q_1$  and  $q_2$  are such that their coordinate-wise difference is at most  $\alpha$  in absolute value, for some  $\alpha \geq 0$ , then the difference of most of the coordinates of  $C(q_1)$  and the corresponding coordinate of  $C(q_2)$  is at most  $\beta$ , for some  $\beta \geq 0$ .
2. (“not close”  $\rightarrow$  “far”): If there exists a coordinate where messages  $q_1$  and  $q_2$  differ by more than  $\alpha'$ , for some  $\alpha' > \alpha$ , then the difference of most of the coordinates of  $C(q_1)$  and the corresponding coordinate of  $C(q_2)$  is more than  $\beta$  in absolute value.

### 2.5.1 Constructing Locality Sensitive Error Correcting Codes

We present a randomized construction of locality sensitive error correcting codes that proceeds in two steps. First, we construct a version of the codes over the alphabet  $\mathbb{Z}$  with weak parameters by taking each coordinate of the codeword to be a random linear function of the coordinates of the messages being encoded (details later). The codes we construct in this step will satisfy the property in item 1 above for an extremely large, say 99.99% of the coordinates but will satisfy the property in item 2 for only 50% of the coordinates.

This is followed by an amplification step where we combine independent copies of the codes constructed in the previous step to get better parameters. Specifically, we take  $L$  copies, for some constant  $L$ , and construct a “joint” codeword each of whose coordinates  $\in \mathbb{Z}^L$  is a tuple consisting of the corresponding coordinate in all the codewords. A pair of such joint coordinates is considered to be far if any one of the constituent coordinates is far. By setting  $L$  to be the right amount, we can make sure that the constant in item 1 only degrades slightly, say to 99%, while the constant in item 2 improves drastically to 99%.

It remains to describe step 1, the construction of un-amplified locality sensitive error correcting codes. As mentioned earlier, every coordinate of these codes is a random linear function of the coordinates of the message being encoded. The coefficients in this linear function belong to  $\{-1, 1\}$  chosen independently and uniformly. A simple concentration argument shows that, for messages  $q_1$  and  $q_2$  of length  $k > 0$  whose coordinate-wise difference is at most  $\alpha$ , the value of such a random linear function will not differ by more than  $\mathcal{O}(\alpha\sqrt{k})$ , with high probability. Thus, setting  $\beta = \mathcal{O}(\alpha\sqrt{k})$  ensures that item 1 above is satisfied.

It remains to show why item 2 is satisfied. For this, assume that two messages  $q_1$  and  $q_2$  of length  $k$  are being encoded such that  $q_1$  and  $q_2$  differ by at least  $\alpha' > \alpha$  in at least one coordinate. Let us assume without loss of generality that this is the last coordinate. Then, for any choice of coefficients for the first  $k - 1$  coordinates, there exists a choice in  $\{-1, 1\}$  of the coefficient for the last coordinate such that the difference in the value of the linear function is at least  $\alpha'$  in magnitude. Indeed, either the difference without the last coordinate is positive, in which we can select the value in  $\{-1, 1\}$  that will make the difference increase by  $\alpha'$ , or it is negative, in which case we can select the other value and make the difference decrease by  $\alpha'$ . In either case, the resulting difference is at least  $\alpha'$  in absolute value.

This implies that, with probability at least  $\frac{1}{2}$ , setting  $\beta = \alpha'$  satisfies item 2 finishing the argument.

## 2.6 Our Protocol

Armed with locality sensitive error correcting codes, we now describe our protocol. Note that our description has parameters such as  $k$ ,  $m$ ,  $\alpha$ , etc. and we set these parameters to appropriate values later in the sketch. Recall that the  $n$  parties are divided into families of size  $A = \Theta(\log n)$  randomly and this ensures that, except with probability polynomially small in  $n$ , all families have a small fraction of malicious parties.

We divide our protocol into the same three stages as in Gallager's protocol, although the order of the stages is different.

- **Stage Broadcast:** In this stage, all the parties broadcast their input  $m$  times (we set  $m$  later in this sketch). The parameter  $m$  will be chosen so that, except with probability at most  $\frac{1}{\text{polylog}(n)}$ , all the parties in a family receive approximately the same counts of the number of 1s from all other parties in the family up to an additive error of  $\alpha$ .

As in Gallager's protocol, as the noise received by each party is independent, except with probability polynomially small in  $n$ , at least 90% of the parties in any family receive the same counts up to an additive error of  $\alpha$ .

- **Stage Boost:** Notwithstanding the high level similarity to it, in that the parties in a family join forces to convince all the parties of the approximately correct counts, stage **Boost** in our protocol is very different from stage **Boost** in Gallager's protocol.

In our protocol, we first divide the families into  $\frac{A}{k}$  groups of  $k$  parties each. Then, for all  $i$ , party  $i$  in the family computes, using a locality sensitive error correcting code, an encoding for each group in the family of the vector of counts it received from that group in stage **Broadcast**, and broadcasts coordinate  $i$  of all the  $\frac{A}{k}$  encodings.

All the parties then combine the coordinates received from the different parties in a family to get a codeword for all the  $\frac{A}{k}$  groups in the family, and decode it to get the individual counts for the parties in the group. We note that these decoded counts are within  $\alpha'$  of the true counts. Indeed, 90% of the parties in the family sent coordinates from encodings of counts that were within  $\alpha$  of the true counts. Only a small constant fraction of coordinates were altered due to corruptions, and another small number of coordinates were sent by malicious parties in each family. Therefore, most of the coordinates received are from encodings of vectors within  $\alpha$  of the true counts. Due to the property in item 1 above, these coordinates are within  $\beta$  of the corresponding coordinates in the encoding of the true counts.

Because of the large number of coordinates that are within  $\beta$  of the corresponding coordinates in the encoding of the true counts, due to item 2 above, these coordinates are unlikely to come from a vector that is not within  $\alpha'$  of the true counts. Consequently, except with probability polynomially small in  $n$ , all the parties decode to a vector of counts within  $\alpha'$  of the true counts, as desired. We will set  $\alpha'$  to be the same order of magnitude as  $m$ , but smaller, say  $\alpha' = \frac{m}{10}$ .

- **Stage Guess:** In this stage, the parties simply perform majority based decoding of the counts decoded after stage **Boost**. This results in correct outputs for non-malicious parties because their true counts are either 0, if they have input 0, or  $m$ , if they have input 1. As the decoded counts are only  $\alpha' = \frac{m}{10}$  off from the true counts, majority based decoding will work correctly for the non-malicious parties.

**Setting the parameters.** Recall the size of a family is  $A = \Theta(\log n)$ . We already explained  $\alpha' = \frac{m}{10}$ , and our construction of locality sensitive error correcting codes in Subsection 2.5 implies  $\beta = \mathcal{O}(\alpha\sqrt{k}) = \frac{m}{10}$  as well. The number of bits communicated by a given party in our protocol is  $m$  in stage **Broadcast** and (roughly)  $\frac{A}{k}$  in stage **Boost**. Thus, total communication by a party equals

$$m + \frac{A}{k} = m + \mathcal{O}\left(A \cdot \frac{\alpha^2}{m^2}\right).$$

In the fault tolerant model, where the best guarantee obtainable is  $\alpha = \Theta(m)$ , the expression above is at least  $\Omega(\log n)$  and we do not get any gains over the protocol in Subsection 2.2. However, in the statistical model, where the more predictable nature of the noise allows us to use concentration bounds and get  $\alpha = \Theta(\sqrt{m})$ , the expression above is minimized when  $m = \Theta(\sqrt{\log n})$ , and we get an improvement over the protocol in Subsection 2.2. Observe that, perhaps surprisingly, the malicious parties are weaker in the model with “more” noise, and this is because the noise is also more predictable.

**Mixability of the encodings.** We finish the section by covering one subtlety about stage Boost that we omitted from the description above. In our implementation, instead of party  $i$  in the family sending coordinate  $i$  of the encoding it computed, we have it send a random coordinate along with the identity of the coordinate. As a typical coordinate will satisfy the properties in item 1 and item 2 above, this does not affect our analysis by much, but it does help us avoid the pathological case where all parties send a coordinate that is atypical in that it does not satisfy item 1 and item 2.

We note that this pathological case never arose in Gallager’s implementation as there, the coordinates sent by different parties in a family were different coordinates from the encodings of the *same* message, and thus, only a small number of them could possibly be atypical. For us, the parties send encodings of *nearby* messages, and it is possible that all coordinates that are sent are atypical for the encoding where they came from, hence, this extra randomization step.

### 3 Models and Formal Problem Definition

#### 3.1 Noisy Copies

Let  $\epsilon \in [0, 1/2)$  be a noise parameter and  $k \in \mathbb{N}$ . An  $\epsilon$ -noise bit is a  $\{0, 1\}$ -valued random variable that takes value 1 with probability *exactly*  $\epsilon$ . An  $\epsilon$ -noise  $k$ -vector  $N$  is a sequence of  $k$  independent  $\epsilon$ -noise bits. For a bit-vector  $x \in \{0, 1\}^k$ , an  $\epsilon$ -noisy copy of  $x$  is a random variable of the form  $x \oplus N$ , where  $\oplus$  denotes the bitwise XOR, and  $N$  is an  $\epsilon$ -noise  $k$ -vector. More generally, if  $X$  is any random variable taking values in  $\{0, 1\}^k$  an  $\epsilon$ -noisy copy of  $X$  is a random variable of the form  $X \oplus N$ , where  $N$  is an  $\epsilon$ -noise  $k$ -vector chosen independently from  $X$ .

#### 3.2 The Noisy Broadcast Model

The (statistical) noisy broadcast model considers  $n$  parties  $P_1, \dots, P_n$ . Let  $\mathcal{X}$  and  $\mathcal{Y}$  be sets. The input is a vector  $x \in \mathcal{X}^n$ , and each of the parties  $P_i$  initially has coordinate  $x^i \in \mathcal{X}$ . The goal is for party  $P_i$  to evaluate a function  $f^i : \mathcal{X}^n \rightarrow \mathcal{Y}$  at  $x$ , i.e., output  $y^i = f^i(x)$ . This goal is to be accomplished by a noisy broadcast protocol.

The specification of a (deterministic) protocol over the noisy broadcast model with noise rate *exactly*  $\epsilon$  consists of:

1. The number  $s \in \{0\} \cup \mathbb{N}$  of broadcasts used in the protocol.
2. A sequence  $i_1, \dots, i_s \in [n]$  of indices of parties (with repetitions allowed).
3. A sequence  $g_1, \dots, g_s$  of broadcast functions, where  $g_j : \mathcal{X} \times \{0, 1\}^{j-1} \rightarrow \{0, 1\}$ .
4. For all  $i$ , an output function  $h^i : \mathcal{X} \times \{0, 1\}^s \rightarrow \mathcal{Y}$  for  $P_i$ .

**Running a protocol.** The execution of a noisy broadcast protocol  $\Pi$  depends on the input  $x$ , and on a noise vector  $N$ . We will think of  $N$  as a concatenation of  $s$  independent  $\epsilon$ -noise  $n$ -vectors  $N_1, \dots, N_s$ . Let  $j \in [s]$ . In the  $j^{\text{th}}$  step of the execution of  $\Pi$ , party  $P_{i_j}$  broadcasts a bit  $b_j$  and all the parties  $i \in [n]$ , receive  $b_j^i = b_j \oplus N_j[i]$ , an independent noisy copy of  $b_j$ . Formally, the bit broadcast by  $P_{i_j}$  at step  $j$  is  $b_j = g_j(x^{i_j}, b_1^{i_j}, b_2^{i_j}, \dots, b_{j-1}^{i_j})$ , that is, the value of  $g_j$  on the input of  $P_{i_j}$  and the  $j-1$  bits received by  $P_{i_j}$  during the first  $j-1$  rounds. The output of  $P_i$  is  $y^i = h^i(x^i, b_1^i, \dots, b_s^i)$ , that is, the value of  $h^i$  on the input of  $P_i$  and the  $s$ -vector of bits received by  $P_i$ .

Note that the assumed model is *non-adaptive* or *oblivious*: the sequence of parties who broadcast is fixed in advance and does not depend on the execution. Without this requirement, the noise could lead to several parties speaking at the same time (a *collision*). Moreover, this problem will be more serious when we introduce malicious parties, who may decide to speak all the time causing collisions in every round. Finally, note that the model rules out *communication by silence*: when it is the turn of a party to speak, it must speak.

### 3.3 The Noisy Broadcast Model with Malicious Parties

So far, we assumed that all participating parties are collaborating and following the protocol. We now consider the model where a subset  $\text{Mal} \subset [n]$  of the parties, called the *malicious parties*, are controlled by an adversary and may not follow the protocol. The set of malicious parties is determined prior to the execution of the protocol and is unchanged throughout the duration of the execution. We consider randomized protocols in this malicious setting and allow parties to use both private randomness (known to a single party) and public randomness (known to all parties). Observe that in the standard (non-malicious) setting, public randomness can always be used in lieu of private randomness. This is no longer possible in our malicious setting, as will be apparent next.

The adversary controlling the malicious parties is assumed to know the inputs of all the parties, the shared random string, and the private random strings of the parties in  $\text{Mal}$ . In addition, in round  $j \in [s]$ , the adversary knows the channel's prior noise vectors  $N_1, N_2, \dots, N_{j-1}$ , and thus also knows all the bits that were sent and received by all the parties in all the previous rounds. (Note that the adversary does not know either the private random strings of the honest parties or the noise in the channel in future rounds). The parties in  $[n] \setminus \text{Mal}$  are still assumed to be following the protocol and are called *honest parties*.

**Computing functions.** Let  $x \in \mathcal{X}^n$  be an input for the parties and let  $\text{Mal}$  be the set of malicious parties. We say that the input  $x' \in \mathcal{X}^n$  is *consistent* with  $x$  and  $\text{Mal}$  if  $\hat{x}^i = x^i$  for every  $i \in [n] \setminus \text{Mal}$ .

Let  $\theta \in [0, 1/2)$  and assume that  $n$  is a sufficiently large function of  $\theta$ . Let  $\delta \geq 0$ . We say that a randomized protocol  $\Pi$  over the noisy broadcast channel with noise rate exactly  $\epsilon$  *computes the functions  $\{f^i\}_{i \in [n]}$  in the presence of  $\theta$ -fraction of malicious parties with error  $\delta$*  if whenever the set of malicious parties  $\text{Mal}$  satisfies  $|\text{Mal}| \leq \theta n$ , then for every  $x \in \mathcal{X}^n$ , with probability at least  $1 - \delta$ , the following holds: For every  $i \in [n] \setminus \text{Mal}$ , there exists  $\hat{x} \in \mathcal{X}^n$  that is consistent with  $x$  and  $\text{Mal}$ , such that the output of  $P_i$  in  $\Pi$  is  $y^i = f^i(\hat{x})$ . Here, the probability is over the private and public randomness and the noise in the channel.

We say that a randomized protocol  $\Pi$  over the noisy broadcast channel with noise rate exactly  $\epsilon$  *computes the identity function in the presence of  $\theta$ -fraction of malicious parties with error  $\delta$*  if  $\mathcal{X} = \{0, 1\}$  and  $\mathcal{Y} = \{0, 1\}^n$  and  $\Pi$  computes the functions  $\{f^i\}_{i \in [n]}$  in the presence of  $\theta$ -fraction of malicious parties with error  $\delta$ , where  $f^i : \{0, 1\}^n \rightarrow \{0, 1\}^n$  is given by  $f^i(x) = x$ . Simplified for the identity function, the above means that whenever  $|\text{Mal}| \leq \theta n$ ,

then, for every input  $x$ , with probability at least  $1 - \delta$ , the following holds: For all  $i \in [n]$ ,  $P_i$  outputs a vectors  $y^i = (\hat{x}_1^i, \hat{x}_2^i, \dots, \hat{x}_n^i)$ , where if  $i, i' \in [n] \setminus \text{Mal}$ , then  $\hat{x}_{i'}^i = x^{i'}$  (otherwise,  $\hat{x}_{i'}^i$  can be arbitrary), as suggested by  $(*)$ .

## 4 Preliminaries

Please refer to the full version for the missing proofs.

### 4.1 Concentration Inequalities

► **Lemma 2** (Multiplicative Chernoff bound). *Suppose  $X_1, \dots, X_n$  are independent random variables taking values in  $[0, 1]$ . Let  $X$  denote their sum and let  $\mu = \mathbb{E}[X]$  denote the sum's expected value. Then,*

$$\begin{aligned} \Pr(X \geq (1 + \delta)\mu) &\leq e^{-\frac{\delta^2 \mu}{2 + \delta}}, & \forall 0 \leq \delta, \\ \Pr(X \leq (1 - \delta)\mu) &\leq e^{-\frac{\delta^2 \mu}{2}}, & \forall 0 \leq \delta \leq 1. \end{aligned}$$

In particular, we have that:

$$\begin{aligned} \Pr(X \geq (1 + \delta)\mu) &\leq e^{-\frac{\delta \mu}{3} \cdot \min(\delta, 1)}, & \forall 0 \leq \delta, \\ \Pr(|X - \mu| \geq \delta \mu) &\leq 2 \cdot e^{-\frac{\delta^2 \mu}{3}}, & \forall 0 \leq \delta \leq 1. \end{aligned}$$

We derive a couple of corollaries of Lemma 2, that will be convenient for us to use.

► **Corollary 3.** *Suppose  $X_1, \dots, X_n$  are independent random variables taking values in  $[0, 1]$ . Let  $X$  denote their sum and let  $\mu = \mathbb{E}[X]$  denote the sum's expected value. Then, for all  $\Delta \geq 2\mu$ , we have:*

$$\Pr(X \geq \Delta) \leq e^{-\frac{\Delta}{6}}.$$

**Proof.** As  $\Delta \geq 2\mu$ , we have  $\Delta = (1 + \delta)\mu$  for some  $\delta \geq 1$ . Applying Lemma 2 with this  $\delta$ , we get  $\Pr(X \geq \Delta) \leq e^{-\frac{\Delta}{3}} \leq e^{-\frac{\Delta}{6}}$ , as desired. ◀

► **Corollary 4.** *Suppose  $X_1, \dots, X_n$  are independent random variables taking values in  $[-1, 1]$  such that  $\mathbb{E}[X_i] = 0$  for all  $i \in [n]$ . If  $X = \sum_{i=1}^n X_i$  denotes their sum, then, for all  $0 \leq \delta \leq 1$ , we have*

$$\Pr(|X| \geq \delta n) \leq 2 \cdot e^{-\frac{\delta^2 n}{6}}.$$

**Proof.** Apply Lemma 2 on the variables  $\frac{X_i + 1}{2}$ . ◀

We shall also use the following version of Chernoff bound for negatively correlated random variables:

► **Definition 5** (Negatively Correlated Random Variables). *For  $n > 0$ , let  $X_1, \dots, X_n$  be random variables that take values in  $\{0, 1\}$ . We say that the random variables  $X_1, \dots, X_n$  are negatively correlated if for all subsets  $S \subseteq [n]$ , we have  $\Pr(\forall i \in S : X_i = 1) \leq \prod_{i \in S} \Pr(X_i = 1)$ .*

► **Lemma 6** (Generalized Chernoff Bound; cf. [14]). *For  $n > 0$ , let  $X_1, \dots, X_n$  be negatively correlated random variables that take values in  $\{0, 1\}$ . Let  $X$  denote their sum and let  $\mu = \mathbb{E}[X]$  denote the sum's expected value. Then, for any  $\delta \geq 0$ , we have:*

$$\Pr(X > (1 + \delta) \cdot \mu) \leq e^{-\frac{\delta^2 \mu}{2 + \delta}}.$$

In particular, we have that:

$$\Pr(X > (1 + \delta)\mu) \leq e^{-\frac{\delta \mu}{3} \cdot \min(\delta, 1)}, \quad \forall \delta \geq 0.$$

## 4.2 Results From Coding Theory

We use the following standard result for error-correcting codes, and include a proof for completeness.

► **Lemma 7.** *Let  $\delta > 0$  and define  $K_0 = \lceil 10/\delta^2 \rceil$ . For all  $n > 0$ , there exists a function  $\text{ECC}_{n,\delta} : \{0,1\}^n \rightarrow \{0,1\}^{K_0 n}$  such that for all  $s \neq t \in \{0,1\}^n$ , we have*

$$\Delta(\text{ECC}_{n,\delta}(s), \text{ECC}_{n,\delta}(t)) > \left(\frac{1}{2} - \delta\right) \cdot K_0 n.$$

## 5 Locality Sensitive Error Correcting Codes

Let  $n > 0$  and  $x, y \in \mathbb{Z}^n$  be vectors. For  $z \in \mathbb{Z}$ , define

$$\text{sep}_z(x, y) = \{i \in [n] \mid |x_i - y_i| > z\}.$$

Observe that  $|\text{sep}_0(\cdot)|$  is simply the Hamming distance between the vectors  $x$  and  $y$ , and is therefore, a metric. For general  $z \in \mathbb{Z}$  however, the function  $|\text{sep}_z(\cdot)|$  may not be a metric. In this paper, we work with the following generalization of the function  $\text{sep}(\cdot)$ .

► **Definition 8.** *Let  $n, L > 0$  and  $x, y \in (\mathbb{Z}^L)^n$  be vectors. For  $z \in \mathbb{Z}$ , define*

$$\text{sep}_z(x, y) = \{i \in [n] \mid \exists l \in [L] : |x_{i,l} - y_{i,l}| > z\}.$$

► **Lemma 9.** *Let  $n, L > 0$  and  $x_1, x_2, y \in (\mathbb{Z}^L)^n$  be vectors. For  $z_1, z_2 \in \mathbb{Z}$ , we have*

$$\text{sep}_{z_1}(x_1, y) = \emptyset \wedge \text{sep}_{z_1+z_2}(x_2, y) \neq \emptyset \implies \text{sep}_{z_2}(x_1, x_2) \neq \emptyset.$$

### 5.1 Definition

We now define locality sensitive error correcting codes. Throughout this section and the next, we fix integers  $m, k > 100$ .

► **Definition 10.** *Let  $n, L > 0$  and  $C : (\{0\} \cup [m])^k \rightarrow (\mathbb{Z}^L)^n$ . Let  $\alpha, \alpha', \beta, \mu, \mu' > 0$  be parameters. We say that the function  $C$  is an  $(n, L, \alpha, \alpha', \beta, \mu, \mu')$ -locality sensitive error correcting code if it has the following two properties:*

■  **$(\alpha, \beta, \mu)$ -locality sensitive:** *For all  $x, y \in (\{0\} \cup [m])^k$ , we have:*

$$\text{sep}_\alpha(x, y) = \emptyset \implies |\text{sep}_\beta(C(x), C(y))| \leq (1 - \mu)n.$$

■  **$(\alpha', \beta, \mu')$ -error correcting:** *For all  $x, y \in (\{0\} \cup [m])^k$ , we have:*

$$\text{sep}_{\alpha'}(x, y) \neq \emptyset \implies |\text{sep}_\beta(C(x), C(y))| \geq \mu'n.$$

We sometimes say  $C$  is a locality sensitive error correcting code or  $C$  is an  $(n, L)$ -locality sensitive error correcting code if we do not wish to emphasize the other parameters. Our protocol requires  $C$  to have a short representation as stated below:

► **Definition 11 (Representation Length).** *Let  $n, L > 0$  and  $C$  be an  $(n, L)$ -locality sensitive error correcting code. Define:*

$$\|C\| = \max_{x \in (\{0\} \cup [m])^k} \max_{i \in [n]} \max_{l \in [L]} |C_{i,l}(x)|.$$

Observe that for all  $x \in (\{0\} \cup [m])^k$ , the value of  $C(x)$ , can be encoded using  $10 \cdot \log(10 \cdot \|C\| + 10)$  bits. We shall use this to show that our protocol does not communicate a lot of bits.

## 5.2 Proof of Existence

We have:

► **Theorem 12.** *Let  $\theta < \frac{1}{2}$  and  $n \geq \left(\frac{100}{1-2\theta}\right)^3 \cdot k \log m$ . There exists a locality sensitive error correcting code  $\mathcal{C}$  with parameters:*

$$\left(n, 10 \cdot \log \frac{10}{1-2\theta}, 4\sqrt{m \log k}, \beta, \beta, \frac{1}{2} + \theta, \frac{1}{2} + \theta\right),$$

where  $\beta = 24\sqrt{mk \cdot \log k \cdot \log \frac{10}{1-2\theta}}$ . Moreover,  $\mathcal{C}$  satisfies  $\|\mathcal{C}\| \leq mk$  and  $\mathcal{C}$  can be computed in time polynomial in  $(m, k, n, L)$ <sup>10</sup>

## 6 Mixability of Codewords

We show that the encodings of similar (but not identical) inputs under a locality sensitive error correcting code can be “mixed” together while maintaining properties similar to the original codewords. In order to formalize this, we first generalize the function  $\text{sep}(\cdot)$  from Definition 8.

► **Definition 13.** *Let  $n, L > 0$ ,  $x \in (\mathbb{Z}^L)^n$ , and  $z \in \mathbb{Z}$ . For  $n' > 0$  and a vector of pairs  $y \in ([n] \times (\mathbb{Z}^L))^{n'}$ , define*

$$\text{sep-mix}_z(x, y) = \{i \in [n'] \mid \exists l \in [L] : |x_{y_{i,1},l} - (y_{i,2})_l| > z\}.$$

Intuitively, every coordinate in  $y$  has two components, the first points to a coordinate in  $x$  and the second one is a value in  $\mathbb{Z}^L$  (a symbol in the code’s alphabet). The function  $\text{sep-mix}_z(\cdot)$  compares each coordinate in  $y$  to the coordinate it points to in  $x$  and checks if they “differ” by more than  $z$ .

An easy corollary of the above definition is the following where for a set  $S \subseteq [n']$ , we use the notation  $y|_S$  to denote the vector  $y$  restricted to the coordinates in  $S$ .

► **Corollary 14.** *Let  $n, L > 0$ ,  $x \in (\mathbb{Z}^L)^n$ , and  $z \in \mathbb{Z}$ . Also, let  $n' > 0$  and  $y \in ([n] \times (\mathbb{Z}^L))^{n'}$ . We have for all  $S \subseteq [n']$  that:*

$$\text{sep-mix}_z(x, y) \cap S = \text{sep-mix}_z(x, y|_S).$$

Fix an  $(n, L, \alpha, \alpha', \beta, \mu, \mu')$ -locality sensitive error correcting code  $\mathcal{C}$  for the rest of this section. We show that:

► **Lemma 15.** *Let  $x \in (\{0\} \cup [m])^k$  and  $n' > 0$ . For  $i' \in [n']$ , let  $y_{i'} \in (\{0\} \cup [m])^k$  be given. If, for  $j_1, j_2, \dots, j_{n'} \in [n]$ , we have*

$$Y(j_1, j_2, \dots, j_{n'}) = ((j_1, \mathcal{C}_{j_1}(y_1)), (j_2, \mathcal{C}_{j_2}(y_2)), \dots, (j_{n'}, \mathcal{C}_{j_{n'}}(y_{n'}))),$$

then, when  $j_1, j_2, \dots, j_{n'}$  are sampled uniformly at random, we have for all  $\Delta_1 \geq 2 \cdot \left(n' - \frac{n'}{n} \cdot \min_{i' \in [n']} |\text{sep}_\beta(\mathcal{C}(x), \mathcal{C}(y_{i'}))|\right)$  that:

$$\Pr\left(n' - |\text{sep-mix}_\beta(\mathcal{C}(x), Y(j_1, j_2, \dots, j_{n'}))| \geq \Delta_1\right) \leq \exp\left(-\frac{\Delta_1}{6}\right).$$

We also have, for all  $\Delta_2 \geq 2 \cdot \frac{n'}{n} \cdot \max_{i' \in [n']} |\text{sep}_\beta(\mathcal{C}(x), \mathcal{C}(y_{i'}))|$  that

$$\Pr\left(|\text{sep-mix}_\beta(\mathcal{C}(x), Y(j_1, j_2, \dots, j_{n'}))| \geq \Delta_2\right) \leq \exp\left(-\frac{\Delta_2}{6}\right).$$

<sup>10</sup> In fact, as  $\mathcal{C}$  will only encode messages of logarithmic length in our protocol, the running time of our protocol will be almost-linear even if encoding  $\mathcal{C}$  took sub-exponential time.



## 7 Our Protocol

We are now ready to state our protocol. Recall that there are  $n$  parties amongst which at most  $\theta n$  are malicious. The following hold:

$$\theta < \frac{1}{2}, \quad \epsilon = \frac{1}{10}, \quad n > 100^{100 \frac{100}{1-2\theta}}. \quad (1)$$

We define the following parameters:

$$\begin{aligned} k &= \frac{\sqrt{\log n}}{\left(\frac{1}{2} - \theta\right)^2} \cdot \frac{1}{\left(\log \frac{10}{1-2\theta}\right)^2}, & m &= 50000k \cdot \left(\log \frac{10}{1-2\theta}\right)^2 \cdot \log \log n, \\ \alpha &= 4\sqrt{m \log k}, & \beta &= 24\sqrt{mk \cdot \log k \cdot \log \frac{10}{1-2\theta}} < \frac{m}{6}, \\ L &= 10 \cdot \log \frac{10}{1-2\theta}, & A &= 10^{20} \cdot \frac{\log n}{\left(\frac{1}{2} - \theta\right)^3} \cdot \frac{4\theta}{\min(4\theta, 1-2\theta)}, \\ \ell &= 10^{10} \cdot \log \left(\frac{\log n}{\frac{1}{2} - \theta}\right). \end{aligned} \quad (2)$$

For the rest of this paper, we reserve  $\mathbf{C} : (\{0\} \cup [m])^k \rightarrow (\mathbb{Z}^L)^A$  to be a locality sensitive error correcting code with parameters:

$$\left(A, L, \alpha, \beta, \beta, \frac{199+2\theta}{200}, \frac{199+2\theta}{200}\right).$$

Such a code is promised by Theorem 12. We have by Definition 10 that, for all  $x, y \in [m]^k$ ,

$$\begin{aligned} \text{sep}_\alpha(x, y) = \emptyset &\implies |\text{sep}_\beta(\mathbf{C}(x), \mathbf{C}(y))| \leq A \cdot \left(\frac{1-2\theta}{200}\right). \\ \text{sep}_\beta(x, y) \neq \emptyset &\implies |\text{sep}_\beta(\mathbf{C}(x), \mathbf{C}(y))| \geq A \cdot \left(\frac{199+2\theta}{200}\right). \end{aligned} \quad (3)$$

Additionally, we reserve ECC to be the one promised by Lemma 7 for  $n \leftarrow \ell$  and  $\delta \leftarrow \frac{1}{10}$ . By Lemma 7, we have  $s \neq t \in \{0, 1\}^\ell$  that:

$$\Delta(\text{ECC}(s), \text{ECC}(t)) > \frac{2}{5} \cdot 1000\ell = 400\ell. \quad (4)$$

### 7.1 Partitioning the Parties

The  $n$  parties are randomly divided into  $n/A$  families of  $A$  parties each. Each family is further divided into  $A/k$  groups of  $k$  parties each. Formally, this division is performed using the following random process: First, sample a permutation uniformly at random from the set  $\mathbf{S}_n$  of permutations on  $[n]$ . Then, for  $a \in [n/A]$ , family  $a$  is the set of parties

$$F_a = \{\sigma(A(a-1) + 1), \sigma(A(a-1) + 2), \dots, \sigma(A(a-1) + A)\}.$$

For  $b \in [A/k]$ , group  $b$  in family  $F_a$  is the set of parties

$$G_{a,b} = \{\sigma(A(a-1) + k(b-1) + 1), \dots, \sigma(A(a-1) + k(b-1) + k)\}$$

As the families are chosen randomly, no family has a lot of malicious parties with high probability. Formally,

► **Lemma 16.** *It holds that:*

$$\Pr_{\sigma \sim \mathbf{S}_n} \left( \exists a \in [n/A] : |F_a \cap \text{Mal}| \geq A \cdot \frac{1+2\theta}{4} \right) \leq \frac{1}{n^{30}}.$$

## 7.2 Our Protocol

For the rest of this section and the analysis, fix a partition of the parties into families such that no family has more than  $\frac{1+2\theta}{4}$  fraction of malicious parties. Due to Lemma 16, this happens except with probability at most  $\frac{1}{n^{30}}$ . We note that this partition allows us to denote a party  $i \in [n]$  using either  $(a, j) \in [n/A] \times [A]$ , emphasizing its family and index within the family, or  $(a, b, c) \in [n/A] \times [A/k] \times [k]$ , emphasizing its family, its group, and the index within the groups. These ways to denote a party  $i$  are equivalent and we use them interchangeably.

Our protocol is symmetric for all the parties and is described in Algorithm 1 from the perspective of party  $i$ . In the algorithm, we sometimes use “partial indexing”, e.g., we write  $q_{a,b'}$  to mean the concatenation of the values  $q_{a,b',c'}$  for all possible values of  $c'$ .

Please refer to the full version for details about the analysis.

■ **Algorithm 1** Our protocol from the perspective of party  $i = (a, j) = (a, b, c)$ .

---

### Stage Broadcast:

- 1: Broadcast input  $x^i$  a total of  $m$  times.
- 2: For  $b' \in [A/k]$  and  $c' \in [k]$ , let  $q_{a,b',c'} \leftarrow$  number of 1s received from party  $(a, b', c')$ .

### Stage Boost:

- 3: For  $b' \in [A/k]$ , sample  $z_{a,b'}$  privately and uniformly from  $[A]$ . Let  $v_{a,b'} \leftarrow C_{z_{a,b'}}(q_{a,b'})$ . Broadcast  $\text{ECC}((z_{a,b'}, v_{a,b'}))$  (note that  $\ell$  bits suffice to encode  $(z_{a,b'}, v_{a,b'})$ ).
- 4: For  $a' \in [n/A]$ ,  $j' \in [A]$ , and  $b'' \in [A/k]$ , decode the  $b''^{\text{th}}$  value received from party  $(a', j')$  to get  $\tilde{C}_{a',b'',j'} = (\tilde{z}_{a',b'',j'}, \tilde{v}_{a',b'',j'})$ .

### Stage Guess:

- 5: For  $a' \in [n/A]$  and  $b' \in [A/k]$ ,  $\tilde{p}_{a',b'} \leftarrow \arg \min_{p' \in (\{0\} \cup [m])^k} |\text{sep-mix}_\beta(C(p'), \tilde{C}_{a',b'})|$ .
  - 6: For  $i' = (a', b', c') \in [n]$ , output  $\tilde{x}_{i'} = \mathbf{1}(\tilde{p}_{a',b',c'} \geq \frac{m}{2})$ .
- 

---

## References

- 1 Noga Alon, Mark Braverman, Klim Efremenko, Ran Gelles, and Bernhard Haeupler. Reliable communication over highly connected noisy networks. In *Symposium on Principles of Distributed Computing (DISC)*, pages 165–173. ACM, 2016.
- 2 Alexandr Andoni and Piotr Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Commun. ACM*, 51(1):117–122, 2008. doi:10.1145/1327452.1327494.
- 3 Alexandr Andoni, Piotr Indyk, and Ilya P. Razenshteyn. Approximate nearest neighbor search in high dimensions. *CoRR*, abs/1806.09823, 2018. arXiv:1806.09823.
- 4 Mark Braverman, Klim Efremenko, Ran Gelles, and Bernhard Haeupler. Constant-rate coding for multiparty interactive communication is impossible. In *Symposium on Theory of Computing (STOC)*, pages 999–1010. ACM, 2016.
- 5 Klim Efremenko, Gillat Kol, and Raghuvansh Saxena. Interactive coding over the noisy broadcast channel. In *Symposium on Theory of Computing (STOC)*, pages 507–520. ACM, 2018.
- 6 Uriel Feige and Joe Kilian. Finding OR in a noisy broadcast network. *Information Processing Letters*, 73(1-2):69–75, 2000.
- 7 Robert G. Gallager. Finding parity in a simple broadcast network. *IEEE Transactions on Information Theory*, 34(2):176–180, 1988.

- 8 Abbas El Gamal. Open problems presented at the 1984 workshop on specific problems in communication and computation sponsored by bell communication research. “*Open Problems in Communication and Computation*”, by Thomas M. Cover and B. Gopinath (editors). Springer-Verlag, 1987.
- 9 Ran Gelles, Amit Sahai, and Akshay Wadia. Private interactive communication across an adversarial channel. *IEEE Trans. Inf. Theory*, 61(12):6860–6875, 2015. doi:10.1109/TIT.2015.2483323.
- 10 Navin Goyal, Guy Kindler, and Michael Saks. Lower bounds for the noisy broadcast problem. *SIAM Journal on Computing*, 37(6):1806–1841, 2008.
- 11 Kokouvi Hounkanli, Avery Miller, and Andrzej Pelc. Global synchronization and consensus using beeps in a fault-prone multiple access channel. *Theoretical Computer Science*, 806:567–576, 2020.
- 12 Eyal Kushilevitz and Yishay Mansour. An  $\omega(d \log(n/d))$  lower bound for broadcast in radio networks. *SIAM J. Comput.*, 27(3):702–712, 1998.
- 13 Ilan Newman. Computing in fault tolerance broadcast networks. In *Computational Complexity Conference (CCC)*, pages 113–122, 2004.
- 14 Alessandro Panconesi and Aravind Srinivasan. Randomized distributed edge coloring via an extension of the chernoff-hoeffding bounds. *SIAM J. Comput.*, 26(2):350–368, 1997. doi:10.1137/S0097539793250767.
- 15 Sridhar Rajagopalan and Leonard J. Schulman. A coding theorem for distributed computation. In *Symposium on the Theory of Computing (STOC)*, pages 790–799, 1994.
- 16 Leonard J Schulman. Communication on noisy channels: A coding theorem for computation. In *Foundations of Computer Science (FOCS)*, pages 724–733. IEEE, 1992.
- 17 Leonard J Schulman. Deterministic coding for interactive communication. In *Symposium on Theory of computing (STOC)*, pages 747–756. ACM, 1993.
- 18 Leonard J. Schulman. Coding for interactive communication. *IEEE Transactions on Information Theory*, 42(6):1745–1756, 1996.



# Sampling Arborescences in Parallel

**Nima Anari**

Stanford University, CA, USA  
anari@cs.stanford.edu

**Nathan Hu**

Stanford University, CA, USA  
zixia314@stanford.edu

**Amin Saberi**

Stanford University, CA, USA  
saber@stanford.edu

**Aaron Schild**

University of Washington, Seattle, WA, USA  
aschild@berkeley.edu

---

## Abstract

We study the problem of sampling a uniformly random directed rooted spanning tree, also known as an arborescence, from a possibly weighted directed graph. Classically, this problem has long been known to be polynomial-time solvable; the exact number of arborescences can be computed by a determinant [33], and sampling can be reduced to counting [18, 16]. However, the classic reduction from sampling to counting seems to be inherently sequential. This raises the question of designing efficient parallel algorithms for sampling. We show that sampling arborescences can be done in RNC.

For several well-studied combinatorial structures, counting can be reduced to the computation of a determinant, which is known to be in NC [9]. These include arborescences, planar graph perfect matchings, Eulerian tours in digraphs, and determinantal point processes. However, not much is known about efficient parallel sampling of these structures. Our work is a step towards resolving this mystery.

**2012 ACM Subject Classification** Theory of computation → Parallel algorithms; Theory of computation → Generating random combinatorial structures; Theory of computation → Random walks and Markov chains

**Keywords and phrases** parallel algorithms, arborescences, spanning trees, random sampling

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.83

**Related Version** *Full Version:* <https://arxiv.org/abs/2012.09502>.

## 1 Introduction

Algorithms for (approximately) counting various combinatorial structures are often based on the equivalence between (approximate) counting and sampling [18, 16]. This is indeed the basis of the Markov Chain Monte Carlo (MCMC) method to approximate counting, which is arguably the most successful approach to counting, resolving long-standing problems such as approximating the permanent [17] and computing the volume of convex sets [13].

Approximate sampling and counting are known to be equivalent for a wide class of problems, including the so-called self-reducible ones [18, 16]. This equivalence is nontrivial and most useful in the direction of reducing counting to sampling. However, for some problems, the “easier” direction of this equivalence, namely the reduction from sampling to counting, proves useful. For these problems, almost by definition, we can count via approaches other than MCMC.



© Nima Anari, Nathan Hu, Amin Saberi, and Aaron Schild;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 83; pp. 83:1–83:18



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

One of the mysterious approaches to counting is via determinant computations. A range of counting problems can be (exactly) solved by simply computing a determinant. A non-exhaustive list is provided below.

- Spanning trees in a graph can be counted by computing a determinant related to the Laplacian of the graph, a result known as the matrix-tree theorem [22].
- Arborescences in a directed graph can be counted by computing a determinant related to the directed Laplacian [33].
- The number of perfect matchings in a planar graph can be computed as the Pfaffian (square root of the determinant) of an appropriately signed version of the adjacency matrix, a.k.a. the Tutte matrix [20].
- The number of Eulerian tours in an Eulerian digraph is directly connected to the number of arborescences, and consequently the determinant related to the directed Laplacian [1, 34].
- Given vectors  $v_1, \dots, v_n \in \mathbb{R}^d$ , the volume sampling distribution on subsets  $S \in \binom{[n]}{d}$  can be defined as follows:

$$\mathbb{P}[S] \propto \det([v_i]_{i \in S})^2.$$

The partition function of this distribution is simply  $\det(\sum_i v_i v_i^T)$  (see, e.g., [10]).

- The number of non-intersecting paths between specified terminals in a lattice, and more generally applications of the Lindström-Gessel-Viennot lemma [24, 14].

Efficient counting for these problems follows the polynomial-time computability of the associated determinants. In turn, one obtains efficient sampling algorithms for all of these problems; we remark that of all of these problems are known to be self-reducible, but nevertheless “easy” slightly varied sampling to counting reductions exist for all of them.

While polynomial-time sampling for all of these problems has long been settled, we reopen the investigation of these problems by considering *efficient parallel sampling* algorithms. We focus on the computational model of PRAM, and specifically on the complexity classes NC and RNC. Here, a polynomially bounded number of processors are allowed access to a shared memory, and the goal is for the running time to be polylogarithmically bounded; the class RNC has additionally access to random bits. Determinants can be computed efficiently in parallel, in the class NC [9], and as a result there are NC counting algorithms for all of aforementioned problems. However, the sampling to counting reductions completely break down for parallel algorithms, as there seems to be an inherent sequentiality in these reductions.

Take spanning trees in a graph as an example. The classic reduction from sampling to counting proceeds as follows:

```

for each edge  $e$  do
     $A \leftarrow$  number of spanning trees containing  $e$ 
     $B \leftarrow$  total number spanning trees
    Flipping a coin with bias  $A/B$ , decide whether  $e$  should be part of the tree.
    Either contract or delete the edge  $e$  based on this decision.
  
```

Each iteration of this loop uses a counting oracle to compute  $A, B$ . However the decision of whether to include an edge  $e$  as part of the tree affects future values of  $A, B$  for other edges, and this seems to be the inherent sequentiality in this algorithm. The sampling-to-counting reduction for all other listed problems encounters the same sequentiality obstacle.

In this paper, we take a step towards resolving the mysterious disparity between counting and sampling in the parallel algorithms world. We resolve the question of sampling arborescences in weighted directed graphs, and as a special case spanning trees in weighted undirected graphs, using efficient parallel algorithms with access to randomness, a.k.a. the class RNC.

We remark that the special case of sampling spanning trees in unweighted undirected graphs was implicitly solved by the work of Teng [30], who showed how to simulate random walks in RNC. When combined with earlier work of Aldous [2] and Broder [6], this algorithm would simulate a random walk on the graph, and from its transcript extract a random spanning tree. However, adding either weights or directions to the graph results in the need for potentially exponentially large random walks, which cannot be done in RNC. Our work removes this obstacle.

► **Theorem 1.** *There is an RNC algorithm which takes a directed graph  $G = (V, E)$  together with edge weights  $w : E \rightarrow \mathbb{R}_{\geq 0}$  as input and outputs a random directed rooted tree  $T$ , a.k.a. an arborescence. The output  $T$  follows the distribution*

$$\mathbb{P}[T] \propto \prod_{e \in T} w(e).$$

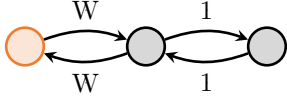
## 1.1 Related Work and Techniques

There is a long line of research on algorithms for sampling and counting spanning trees and more generally arborescences. The matrix-tree theorem of Kirchhof [22] showed how to count spanning trees in undirected graphs, and later Tutte [33] generalized this to arborescences in digraphs. Somewhat surprisingly Aldous [2] and Broder [6] showed that random spanning trees and more generally random arborescences of a graph can be extracted from the transcript of a random walk on the graph itself. The main focus of subsequent work on this problem has been on improving the total running time of sequential algorithms for sampling. After a long line of work [35, 7, 21, 25, 12, 11], Schild [28] obtained the first almost-linear time algorithm for sampling spanning trees. More recently Anari, Liu, Oveis Gharan, and Vinzant [3] improved this to nearly-linear time. Many of these works are based on speeding up the Aldous-Broder algorithm. Our main result, Theorem 1, is also built on the Aldous-Broder algorithm, but we focus on parallelizing it instead of optimizing the total running time.

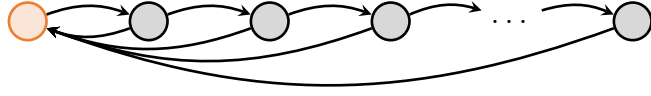
No almost-linear time algorithm is yet known for sampling arborescences in digraphs, as opposed to spanning trees in undirected graphs. While sampling spanning trees has a multitude of application (see [28]), there are a number of applications for the directed graph generalization. Most notably, there is a many-to-one direct correspondence between Eulerian tours in an Eulerian digraph and arborescences of the graph. This correspondence, known as the BEST theorem [1, 34] allows one to generate random Eulerian tours by generating random arborescences (see [8]). We leave the question of whether the correspondence in the BEST theorem is implementable in NC to future work, but note that sampling Eulerian tours has interesting applications in biology and sequence processing [19, 27]. We remark that, unlike directed graphs, generating random Eulerian tours of *undirected* Eulerian graphs in polynomial time is a major open problem [32].

In a slightly different direction related to this work, Balaji and Datta [4] considered the space complexity of counting arborescences. They showed this problem is in L for graphs of bounded tree-width, obtaining algorithms for counting Eulerian tours in these digraphs as well.





■ **Figure 1** Starting from left it takes  $\Theta(W)$  steps to cover.



■ **Figure 2** A random walk started from the left node covers all  $n$  nodes in time  $\Theta(2^n)$ .

Perhaps in the first major result of its kind in the search for RNC sampling algorithms, Teng [31] showed implicitly how to sample spanning trees in undirected graphs in RNC. Teng [31] showed how to parallelize the simulation of a random walk on a graph; more precisely, he showed how to output a length  $L$  trace of a walk on size  $n$  Markov chains in parallel running time  $\text{polylog}(L, n)$  using only  $\text{poly}(L, n)$  many processors. The Aldous-Broder algorithm extracts an arborescence from the trace of a random walk by extracting the so-called first-visit edge to each vertex. If a random walk is simulated until all vertices are visited at least once, the trace of the random walk has enough information to extract all such first-visit edges. This allows RNC sampling of arborescences in graphs where the number of steps needed to visit all vertices, known as the cover time, is polynomially bounded. While the cover time of a random walk on an undirected unweighted graph is polynomially bounded in the size of the graph (see, e.g., [23]), adding either weights or directions can make the cover time exponentially large. See Figures 1 and 2.

We overcome the obstacle of exponentially large cover times, by taking a page from some of the recent advances on the sequential sampling algorithms for spanning trees. Instead of simulating the entirety of the random walk until cover time, we extract only the first-visit edges to each vertex. We use the same insight used in several prior works that once a region of the graph has been visited, subsequent visits of the random walk can be *shortcut* to the first edge that exits this region. However, this involves a careful construction of a hierarchy of “regions”, and bounding the number of steps needed inside each region to fully cover it. Unlike undirected graphs, in directed graphs arguing about the covering time of a region becomes complicated. We build on some of the techniques developed by Boczkowski, Peres, and Sousi [5] to bound these covering times in the case of Eulerian digraphs. We then reduce the arborescence sampling problem for arbitrary digraphs to that of Eulerian digraphs.

## 1.2 Overview of the Algorithm

At a high-level, our algorithm first proceeds by reducing the problem to sampling an arborescence from an Eulerian graph. We then construct a “loose decomposition” of the Eulerian graph into weighted cycles. We then construct a hierarchy of vertex sets by starting with the empty graph and adding the weighted cycles one by one, from the lowest weight to the highest, adding the connected components in each iteration to the hierarchy. This results in a hierarchical clustering of vertices, with the intuitive property that once we enter a cluster, we spend a lot of time exploring inside before exiting. Our algorithm proceeds by “simulating” polynomially many jumps between the children of each cluster, before exiting that cluster. The resulting jumps are then stitched together at all levels of the hierarchy to form a partial subset of the transcript of a random walk.

The shallow “simulation” of edges jumping across children of a cluster in the hierarchy can be done in RNC by using a doubling trick as in the work of Teng [31]; in order to “simulate”  $L$  jumping edges, we combine the first  $L/2$  with the last  $L/2$  in a recursive fashion. A naive

implementation of this is not parallelizable however, as one needs to know *where* the first  $L/2$  jumps eventually land before “simulating” the rest. However since the number of locations is polynomially bounded, one can precompute an answer for each possible landing location in parallel to simulating the first  $L/2$  edges. Once shallow jumps are simulated, our algorithm then proceeds by stitching these jumping edges with jumping edges of one level lower (i.e., inside of the children clusters), and so on. Again, a naive implementation of this stitching is not in RNC, but we employ a similar doubling trick and an additional caching trick to parallelize everything. We then argue that with high probability all of the edges extracted by this algorithm contain the first-visit edges to every vertex.

## 2 Preliminaries

We use the notation  $[n]$  to denote the set  $\{1, \dots, n\}$ . We use the notation  $\tilde{O}(\cdot)$  to hide polylogarithmic factors. When we use the term high probability, we mean with probability  $1 - \text{poly}(\frac{1}{n})$ , where  $n$  is the size of the input, and the polynomial can be taken arbitrarily large by appropriately setting parameters.

### 2.1 Graph Theory Notations

When a graph  $G = (V, E)$  is clear from the context, we use  $n$  to refer to  $|V|$  and  $m$  to refer to  $|E|$ .

For a subset of vertices  $S \subset V$ ,  $\delta^+(S)$  denotes the set all incoming edges  $\{e = (u, v) \mid v \in S, u \notin S\}$ . Similarly,  $\delta^-(S)$  will denote the set of all outgoing edges  $\{e = (u, v) \mid u \in S, v \notin S\}$ . Lastly, let  $G(S)$  denote the subgraph induced by  $S$ .

► **Definition 2.** *Given a weighed graph  $G = (V, E)$  with weights  $w : E \rightarrow \mathbb{R}_{\geq 0}$ , a subset of the vertices  $S \subseteq V$  is said to be strongly connected by edges of weight  $w_c$  if for every  $s, t \in S$ , there exists a path of edges inside  $S$  connecting  $s$  to  $t$  such that every edge  $e$  of the path has weight  $w(e) \geq w_c$ .*

► **Definition 3.** *An arborescence is a directed graph rooted at vertex  $r$  such that for any other vertex  $v$ , there is exactly one path from  $v$  to  $r$ .*

Arborescences are also known as directed rooted trees and are a natural analog of spanning trees in directed graphs. When a background graph  $G = (V, E)$  is clear from context, by an arborescence we mean a subgraph of  $G$  that is an arborescence.

Given digraph  $G = (V, E)$  together with a weight function  $w : E \rightarrow \mathbb{R}_{> 0}$ , the random arborescence distribution is the distribution that assigns to each arborescence  $T$  probability

$$\mathbb{P}[T] \propto \prod_{e \in T} w(e).$$

When  $w$  is not given, it is assumed to be the constant 1 function, and the random arborescence distribution becomes uniformly distributed over all arborescences. We view an undirected graph  $G = (V, E)$  as a directed graph with double the number of edges, with each undirected edge producing two directed copies in the two possible directions. It is easy to see that the weighted/uniform random arborescence distribution on the directed version of an undirected graph is the same as the weighted/uniform spanning tree distribution on the original graph; viewing each arborescence without directions yields the corresponding spanning tree.

We call a (possibly weighted) digraph  $G = (V, E)$  Eulerian if

$$\sum_{e \in \delta^+(v)} w(e) = \sum_{e \in \delta^-(v)} w(e)$$

for all vertices  $v$ . Note that the directed version of an undirected graph is automatically Eulerian.

## 2.2 Random Walks and Arborescences

Our approach centers around the Aldous-Broder algorithm on directed graphs [2, 6]. Their work reduces the task of randomly generating a spanning tree or arborescence to simulating a random walk on the graph until all vertices have been visited. While most famously known for generating spanning trees, this algorithm can be used to generate arborescences as well.

A weighted digraph defines a natural random walk. This is a Markov process  $X_0, X_1, \dots$ , where each  $X_i$  is obtained as a random neighbor of  $X_{i-1}$  by choosing an outgoing edge with probability proportional to its weight, and transitioning to its endpoint.

$$\mathbb{P}[X_i \mid X_{i-1}] \propto w(X_{i-1}, X_i).$$

A stationary distribution  $\pi$  is a distribution on the vertices such that if  $X_0$  is chosen according to  $\pi$ , then all  $X_i$  are distributed as  $\pi$ . Under mild conditions, namely strong connectivity and aperiodicity, the stationary distribution is unique.

A random walk on an undirected graph is known to be time-reversible. That is if  $X_0$  is started from the stationary distribution, then  $(X_i, X_{i+1})$  is identical in distribution to  $(X_{i+1}, X_i)$ . This does not hold for directed graphs. However the time reversal of the process  $\dots, X_i, X_{i+1}, \dots$  corresponds to a random walk on a different digraph.

► **Definition 4.** For a weighted digraph  $G = (V, E)$  with weights  $w : E \rightarrow \mathbb{R}_{\geq 0}$  and stationary distribution  $\pi$ , define the time-reversal to be the graph  $G' = (V, E')$  on the same set of vertices, with edges reversed in direction, and weights given by

$$w'(v, u) = \pi(u)w(u, v)/\pi(v).$$

The random walk on  $G'$  shares the same stationary distribution  $\pi$  as  $G$ . The random walk on  $G'$  (started from  $\pi$ ) is identical in distribution to the time-reversed random walk on  $G$  (started from  $\pi$ ); see [23].

► **Theorem 5** ([2, 6]). Suppose that  $G = (V, E)$  is a strongly connected weighted graph, with weights given by  $w : E \rightarrow \mathbb{R}_{> 0}$ . Perform a time-reversed random walk, starting from a vertex  $r$ , until all vertices are visited at least once. For each vertex  $v \in V \setminus \{r\}$ , record the edge used in the random walk to reach  $v$  for the first time. Let  $T$  be the collection of all these first-visit edges (with directions reversed). Then  $T$  is an arborescence of  $G$  rooted at  $r$ , and

$$\mathbb{P}[T] \propto \prod_{e \in T} w(e)$$

This allows us to sample  $r$ -rooted arborescence from a digraph by performing a random walk on the time-reversal. It is also known that among all arborescences, the total weight of those rooted at  $r$  is proportional to  $\pi(r)$ , where  $\pi$  is the stationary distribution [2, 6]. Therefore to resolve Theorem 1, it is enough to show how to sample  $r$ -rooted arborescences.

► **Theorem 6.** There is an RNC algorithm which takes a directed graph  $G = (V, E)$  together with edge weights  $w : E \rightarrow \mathbb{R}_{\geq 0}$  and a root vertex  $r \in V$  as input and outputs a random  $r$ -rooted arborescence, where

$$\mathbb{P}[T] \propto \prod_{e \in T} w(e).$$

We can fix the root because the stationary distribution  $\pi$  can be computed in NC, by solving a system of linear equations.

## 2.3 Parallel Algorithms

In this paper we consider parallel algorithms run on the PRAM model, and we will construct algorithms to show that sampling problems are in the class RNC, the randomized version of NC. In this class, we are allowed to use polynomially (in the input size) many processors who share access to a common random access memory, and also have access to random bits. The running time of our algorithms must be polylogarithmic in the input size.

While our work hinges upon the simulation of a random walk, a process which is inherently sequential, for polynomially many steps, such a task is known to be in RNC through the use of a “doubling trick.”

► **Theorem 7** ([31]). *Suppose that  $G = (V, E)$  is a directed graph, with weights given by  $w : E \rightarrow \mathbb{R}_{\geq 0}$ . The transcript of a random walk starting from a given vertex  $v_0 \in V$  and running for  $L$  time steps can be produced in the PRAM model using  $\text{poly}(|E|, L)$  processors in  $\text{polylog}(|E|, L)$  time.*

Theorem 5 and Theorem 7 naturally lead to a parallel algorithm that randomly samples arborescences using  $\text{poly}(|E|, L)$  processors and  $\text{polylog}(|E|, L)$  time where  $L$  is the cover time of the digraph. However,  $L$  can be exponential in the size of the graph as the cover time depends closely on the edge weights of a graph. In directed graphs, the cover time can be exponential even if we do not allow weights. See Figures 1 and 2.

Luckily, Theorem 5 only needs the first-visit edges to produce the random arborescence. This insight has been heavily used to improve the running time of sequential algorithms for sampling spanning trees [21, 25, 28], by *shortcutting* the random walk. Here we use the same insight to design an RNC algorithm. Our algorithm identifies a hierarchy of clusters of vertices  $S \subseteq V$  and will only simulate incoming and outgoing edges of each cluster as opposed to the entire walk. To do this, we will use a well-known primitive that the probability of entering or exiting a cluster through any edge can be computed using a system of linear equations.

► **Lemma 8** (see, e.g., [28]). *Given a set  $S \subset V$  and vertex  $v \in S$ , the probability of a random walk started at  $v$  exiting  $S$  through any particular edge  $e \in \delta^-(S)$  can be computed by solving a system of linear equations involving the Laplacian.*

Some of the most powerful primitives for NC algorithms come from linear algebra. In particular, multiplying matrices, computing determinants, and inverting matrices all have NC algorithms [9]. Combining this with Lemma 8 we obtain the following folklore result.

► **Lemma 9.** *Given a set  $S \subseteq V$  and vertex  $v \in S$ , the probability of a random walk started at  $v$  exiting  $S$  through any particular edge  $e \in \delta^-(S)$  can be computed in NC.*

## 2.4 Schur Complements

A key tool which we shall use in our analysis of random walks is Schur complements.

► **Definition 10.** *For any weighted digraph  $G = (V, E)$ , for any subset of the vertices  $S \subseteq V$ , we define the graph  $G_S$  to be the Schur complement of  $\bar{S}$ .  $G_S$  is the directed graph formed by starting with the induced sub-graph of  $S$ . Then, for every pair of vertices  $(u, v)$  in  $S$ , add an edge from  $u$  to  $v$  with weight  $\deg(u) \mathbb{P}[u \rightarrow v]$ , where  $\mathbb{P}[u \rightarrow v]$  is the probability that a random walk on  $G$  currently at  $u$  exits  $S$  with the next move and its first return to  $S$  is at the vertex  $v$ .*

Then, it is easy to see that for any starting vertex  $v_0 \in S$ , the distribution over random walk transcripts in  $G_S$  starting at  $v_0$  is the same as the distribution over random walk transcripts in  $G$  starting at  $v_0$  with all vertices outside of  $S$  removed. We also observe that:

► **Proposition 11.** *For any Eulerian weighted digraph  $G = (V, E)$ , for any subset of the vertices  $S \subset V$ , the Schur complement  $G_S$  is also an Eulerian weighted digraph and the degree of any vertex in  $S$  is the same in  $G_S$  as in  $G$ .*

### 3 Reduction to Eulerian Graphs

A key part of our algorithm is the analysis of the time it takes for random walks to cover regions of a digraph. Cover times are in general difficult to analyze on directed graphs, but the situation becomes much easier on Eulerian graphs. For more precise statements, see Section 6. As a first step in our algorithm, we show how to reduce the design of sampling algorithms on arbitrary digraphs to Eulerian ones.

We can reduce the problem of sampling an arborescence of digraph  $G$  to sampling a random arborescence rooted at vertex  $r$  on strongly connected Eulerian digraph  $G''$  as follows in Algorithm 1. We begin by selecting the vertex at which the arborescence is rooted, using the Markov chain tree theorem [2, 6].

► **Theorem 12** (The Markov chain tree theorem). *Let  $G = (V, E)$  be a weighted directed graph. Assume that the natural Markov chain associated with  $G$  has stationary distribution  $\pi = (\pi_1, \dots, \pi_n)$ . Then we have the following relationship between  $\pi$  and the arborescences of  $G$ :*

$$\pi_i = \frac{\sum \{w(T) \mid T \text{ arborescence rooted at } i\}}{\sum \{w(T) \mid T \text{ arborescence}\}},$$

where  $w(T)$  is product of edge weights in  $T$ .

Finding the stationary distribution of a random walk on some graph  $G$  reduces to solving a system of  $n$  linear equations and can be done in NC.

After the root  $r \in V$  of an arborescence is selected, we can add outgoing edges from  $r$  to graph  $G$  to guarantee that  $G$  is strongly connected. As no arborescence rooted at  $r$  will contain these new edges, the addition of such edges does not affect the distribution of arborescences rooted at  $r$ . Then, we note that for any vertex  $v$ , if we multiply the weights of all edges in  $\delta^-(v)$  by some constant, the distribution of arborescences rooted at any vertex does not change. We can therefore rescale edges to ensure that the resulting graph is Eulerian.

► **Proposition 13.** *Graph  $G''$  is Eulerian.*

**Proof.** We have that, for all  $v$  in  $G'$

$$\sum_{(u,v) \in \delta^+(v)} \pi'(u) \frac{w'(u,v)}{\deg'_{\text{out}}(u)} = \pi'(v)$$

Then, considering  $G''$  :

$$\begin{aligned} \deg''_{\text{in}}(v) &= \sum_{(u,v) \in \delta^+(v)} w''(u,v) = \sum_{(u,v) \in \delta^+(v)} \frac{w'(u,v)\pi'(u)}{\deg'_{\text{out}}(u)} = \pi'(v) \\ &= \pi'(v) \sum_{(v,x) \in \delta^-(v)} \frac{w'(v,x)}{\deg'_{\text{out}}(v)} = \sum_{(v,x) \in \delta^-(v)} w''(v,x) = \deg''_{\text{out}}(v) \end{aligned} \quad \blacktriangleleft$$

---

**Algorithm 1** Reduction to strongly-connected Eulerian graphs.
 

---

```

Compute the stationary distribution  $\pi(v)$  of a random walk on  $G$ 
Choose vertex  $r$  to be the root of the arborescence with probability  $\pi(r)$ 
 $G' = (V, E') \leftarrow G$ 
for  $v \in V \setminus \{r\}$  in parallel do
  if  $e' = (r, v) \notin E$  then
    Add  $e'$  to  $E'$  with weight  $w'(e) = 1$ 
Compute the stationary distribution  $\pi'(v)$  of a random walk on  $G'$ 
 $G'' = (V, E'') \leftarrow G'$ 
for  $v \in V$  do
  for  $e = (v, u) \in E''$  do
     $w''(e) \leftarrow \frac{w'(e)}{\deg_{out}(v)} \pi'(v)$ 
Randomly sample an arborescence rooted at  $r$  from  $G''$ 

```

---

Thus, in RNC, the task of randomly sampling an arborescence in some digraph can be reduced to the task of randomly sampling an arborescence rooted at some vertex  $r$  from a strongly connected Eulerian digraph.

## 4 Cycle Decomposition

Because Theorem 5 only needs the first-visit edges to produce a random arborescence, our main idea is to create a hierarchical clustering of the graph that we call  $\mathcal{T}$ . Every element  $\mathcal{S} \in \mathcal{T}$  in the clustering is a subset of  $V$  which is strongly connected by edges of relatively high weight compared to the edges in  $\delta^-(\mathcal{S})$ . Intuitively, a random walk will spend much time traversing edges inside a cluster, before venturing out. Aside from first visit edges, any other traversals do not need to be simulated, thus much of the work can be avoided by only simulating the first few edges which enter or exit a cluster. Consider the pseudocode in Algorithm 2 for generating one such decomposition:

---

**Algorithm 2** Generating a hierarchical decomposition.
 

---

```

for edge  $e \in G$  in parallel do
  Find a cycle  $C_e$  of edges such that  $e \in C_e$  and for every  $e' \in C_e$ ,  $w_{e'} \geq \frac{w_e}{m}$  using
    Algorithm 3
Sort the cycles found by minimum edge weight in decreasing order  $C_1, C_2, \dots, C_m$ 
 $\mathcal{T} \leftarrow \{\}$ 
for  $i = 1, 2, \dots, m$  in parallel do
  Find  $\mathcal{S}_{i,1}, \mathcal{S}_{i,2}, \dots, \mathcal{S}_{i,j}$ , the vertex sets of the connected components in
     $(V, C_1 \cup C_2 \cup \dots \cup C_i)$ 
  Add all elements found to  $\mathcal{T}$ 
for  $\mathcal{S}_{i,j} \in \mathcal{T} \setminus \mathcal{S}_{m,1}$  in parallel do
  Find the node of the form  $\mathcal{S}_{i+1,k}$  such that  $\mathcal{S}_{i,j} \subset \mathcal{S}_{i+1,k}$  and assign  $\mathcal{S}_{i+1,k}$  as the
    parent of  $\mathcal{S}_{i,j}$ 
Contract duplicate nodes in  $\mathcal{T}$ 

```

---

■ **Algorithm 3** FindCycle( $e = (e_0, e_1), G = (E, V)$ ).

---

```

Construct  $G' = (V, E')$ , where  $E' = \{e' \in E : w_{e'} \geq \frac{w_e}{m}\}$ 
for  $v_1, v_2 \in V$  in parallel do
    if  $(v_1, v_2) \in E'$  then
         $R(v_1, v_2, 1) \leftarrow \text{True}$ 
for  $v \in V$  in parallel do
     $R(v, v, 0) \leftarrow \text{True}$ 
     $R(v, v, 1) \leftarrow \text{True}$ 
for  $l \in \{2, 4, \dots, 2^{\lceil \ln n \rceil}\}$  do
    for  $v_1, v_2, v_3 \in V$  in parallel do
        if  $R(v_1, v_2, l/2)$  and  $R(v_2, v_3, l/2)$  then
             $R(v_1, v_3, l) \leftarrow \text{True}$ 
 $c_0 \leftarrow e_1, c_{2^{\lceil \ln n \rceil}} \leftarrow e_0$ 
for  $l \in \{2^{\lceil \ln n \rceil - 1}, 2^{\lceil \ln n \rceil - 2}, \dots, 1\}$  do
    for  $i \in [2^{\lceil \ln n \rceil}]$  such that  $l$  is the largest power of 2 which divides  $i$  in parallel
        do
             $\text{Set } c_i \text{ to be some vertex such that } R(c_{i-l}, c_i, l) \text{ and } R(c_i, c_{i+l}, l)$ 
Prune the transcript of vertices  $\{c_0, c_1, \dots, c_{2^{\lceil \ln n \rceil}}\}$  to remove any duplicates and
return the resulting cycle.

```

---

We use Algorithm 3 to construct a cycle of comparably high-weight edges containing a given edge  $e$ . The collection of these cycles is a “loose decomposition” of the Eulerian graph into cycles. While an Eulerian graph can be decomposed into cycles in general, we do not know if this can be done in NC. Instead we find a collection of cycles whose sum and  $\text{poly}(m)$  times it sandwich the Eulerian graph.

For each edge  $e$ , Algorithm 3 successfully returns a cycle since,

► **Proposition 14.** *In directed weighted Eulerian graph  $G = (V, E)$  for any edge  $e \in E$ , there exists a cycle of edges  $C_e$  such that  $e \in C_e$  and for all  $e' \in C_e, w_{e'} \geq \frac{w_e}{m}$*

**Proof.** Note that any Eulerian digraph can be decomposed into at most  $m$  cycles where each cycle contains edges of uniform weight. Then, for any edge  $e$ , one of these cycles must have weight at least  $\frac{w_e}{m}$ . ◀

Given that a cycle is found containing every edge in  $G$ , it follows that Algorithm 2 successfully generates a hierarchical decomposition which satisfies:

► **Proposition 15.**  *$\mathcal{T}$  is a laminar family, i.e. has the following properties:*

1. Every node  $\mathcal{S}$  of  $\mathcal{T}$  is a subset of  $V$  and is the vertex set of a connected component of  $G$ .
2. Children of any node in  $\mathcal{T}$  are proper subsets of it.
3.  $|\mathcal{T}| \leq 2n$ .

Given this clustering  $\mathcal{T}$ , for any  $\mathcal{S} \in \mathcal{T}$ , we say an edge  $e \in E$  jumps between children of  $\mathcal{S}$  if  $\mathcal{S}$  is the lowest node in  $\mathcal{T}$  which contains both endpoints of  $e$ . We call  $e$  a *jumping edge* of  $\mathcal{S}$ .

In addition, we note that

► **Lemma 16.** *For any  $\mathcal{S} \in \mathcal{T}$ , let  $w_{\max}(\mathcal{S})$  be the maximum weight among the jumping edges of  $\mathcal{S}$ .  $G(\mathcal{S})$  is strongly connected by edges of weight  $\frac{w_{\max}(\mathcal{S})}{m}$ .*



**Proof.** We prove by contradiction. Assume that there is some edge  $e$  jumping between children of  $\mathcal{S}$  and two vertices  $u, v \in \mathcal{S}$  such that every path between  $u, v$  contains an edge of weight less than  $\frac{w_e}{m}$ . Let the cycles which connect  $\mathcal{S}$  in Algorithm 2 be  $C_1, \dots, C_k$ , with  $C_k$  being the cycle with an edge of minimum weight. Then, consider the path of edges connecting  $u$  to  $v$  only containing edges in these cycles. The weight of every edge in the path must be greater than  $w(C_k)$ , the weight of the minimum weight edge in  $C_k$ . Thus by our assumption we must have that  $\frac{w_e}{m} > w(C_k)$ .

But now consider  $C_e$ , the cycle found containing edge  $e$  with minimum edge weight  $w(C_e) \geq \frac{w_e}{m}$ . As  $w(C_e) > w(C_k)$ , this cycle must have been used to construct a child of  $\mathcal{S}$  in  $\mathcal{T}$ . This implies the  $e$  is contained in a child of  $\mathcal{S}$  and is therefore not a jumping edge of  $\mathcal{S}$ , a contradiction.  $\blacktriangleleft$

Lemma 16 intuitively states that the nodes in  $\mathcal{T}$  are tightly connected by edges of relatively high weight compared to incoming or outgoing edges. This will be critical in bounding the number of edges which need to be simulated.

## 5 Parallel Sampling

For clarity, a partially sampled random walk shall be represented as an alternating series of clusters and edges:  $\mathcal{S}_1, e_1, \mathcal{S}_2, e_2, \dots$ , where each  $\mathcal{S}_i \in \mathcal{T}$  and  $e_i \in \delta^-(\mathcal{S}_i) \cap \delta^+(\mathcal{S}_{i+1})$ . The sequence has a natural interpretation of a random walk in  $\mathcal{S}_i$  which then exits  $\mathcal{S}_i$  through  $e_i$  to continue in  $\mathcal{S}_{i+1}$  before traversing  $e_{i+1}$  and so on.

Once we have a hierarchical decomposition  $\mathcal{T}$  found in Section 4, a natural way to use such a decomposition is to simulate a random walk as follows:

1. Simulate just the edges jumping between high-level clusters of a random walk, producing a transcript of the form  $\mathcal{S}_1, e_1, \mathcal{S}_2, e_2, \dots$ .
2. For each  $i$ , recursively simulate a random walk on  $\mathcal{S}_i$  conditioned on exiting  $\mathcal{S}_i$  through edge  $e_i$ .

While this recursive procedure as described is not in RNC, using doubling tricks and memoization, we show how to perform this in RNC. One can view this as a generalization of the doubling trick used by Teng [31].

### 5.1 Simulating Jumping Edges

To simulate edges jumping across children of a node, we generalize the techniques of [31] to the following:

► **Theorem 17.** *Suppose that  $G = (V, E)$  is a directed graph, with weights given by  $w : E \rightarrow \mathbb{R}_{\geq 0}$ . Let  $\mathcal{S} \subseteq V$  be the disjoint union of  $\mathcal{S}_1, \mathcal{S}_2, \dots, \mathcal{S}_k$ , where  $\mathcal{S}$  and each  $\mathcal{S}_i$  is strongly connected. The transcript of first  $L$  edges jumping across different  $\mathcal{S}_i$  in a random walk starting from a given vertex  $v_0 \in \mathcal{S}$  and conditioned on exiting  $\mathcal{S}$  through some edge  $e \in \delta^-(\mathcal{S})$  can be produced in the PRAM model using  $\text{poly}(|E|, L)$  processors and  $\text{polylog}(|E|, L)$  time.*

**Proof.** We denote this function  $\text{JumpingEdges}(\mathcal{S}, v_0, e, L)$ . See Algorithm 4 for pseudocode. Note that we can condition a walk on  $\mathcal{S}$  to exit via some edge  $e \in \delta^-(\mathcal{S})$  by considering a random walk on the graph induced by  $\mathcal{S}$  with an additional edge  $e$  and an absorbing state at the endpoint of  $e$ . For any vertex  $v \in \mathcal{S}$ , let  $\mathcal{S}_v$  denote the unique  $\mathcal{S}_i$  which contains  $v$ .

From Lemma 9, for any  $v \in \mathcal{S}$ , the probability that a random walk currently at  $v$  exits  $\mathcal{S}_v$  through each edge in  $\delta^-(\mathcal{S}_v)$  can be computed in NC. Thus, after using  $\text{poly}(|E|, L)$  processors and  $\text{polylog}(|E|, L)$  time to compute all such probabilities for every vertex  $v \in \mathcal{S}$ ,

### 83:12 Sampling Arborescences in Parallel

we can assume access to a deterministic function  $\text{Next}(v, X)$  which takes a vertex  $v \in \mathcal{S}$  and number  $X \in [0, 1]$  and outputs an edge  $e \in \delta^-(\mathcal{S}_v)$  such that if  $X$  is chosen uniformly from  $[0, 1]$ , then the probability that  $\text{Next}(v, X)$  returns some edge  $e \in \delta^-(\mathcal{S}_v)$  is the probability a random walk currently at  $v$  exits  $\mathcal{S}_v$  through edge  $e$ .

Given this function, if we generate independent uniformly random numbers  $X_1, \dots, X_L \in [0, 1]$ , the transcript of the jumping edges in random walk can be extracted by starting at  $v = v_0$  and repeatedly applying  $\text{Next}(v, X_i)$  to get the next vertex  $v$ , for  $i \in [L]$ . Of course, a naive implementation does not leverage parallelism, so we instead use a doubling trick.

We compute the values  $\text{End}(v, t, l)$  which would be the final edge jumping between children of  $\mathcal{S}$  when we run a random walk starting at  $v$ , and applying  $\text{Next}$  with random inputs  $X_t, X_{t+1}, \dots, X_{t+l-1}$ . For  $l = 1$ , these are 1-step random walks and we use  $\text{Next}$  to compute them. For simplicity, we allow the first argument of  $\text{End}$  to also be an edge, in which case the random walk starts at the ending vertex of that edge. Then we use the following identity, which allows us to compute  $\text{End}$  values for a particular  $l$  from  $\text{End}$  values for  $\frac{l}{2}$ .

$$\text{End}(v, t, l) = \text{End}(\text{End}(v, t, l/2), t, l/2)$$

This is because we can break an  $l$ -step random walk into two  $l/2$ -step random walks. So we can compute all  $\text{End}$  values when  $l$  ranges over powers of 2, in  $\log(L)$  steps.

Finally to compute the edge taken by the random walk at any time  $l = 1, \dots, L$ , we simply write down  $l$  as a sum of powers of 2, and repeatedly use the  $\text{End}$  function to compute the  $l$ -th edge of the random walk which jumps between children of  $\mathcal{S}$ . ◀

---

■ **Algorithm 4**  $\text{JumpingEdges}(\mathcal{S}, v_0, e_{\text{end}}, L)$ .

---

```

for  $t \in \{1, 2, \dots, L\}$  in parallel do
    Let  $X_t$  be a uniformly random number sampled from  $[0, 1]$ 
    for  $v \in \mathcal{S}$  in parallel do
         $\text{End}(v, t, 1) \leftarrow \text{Next}(v, X_t)$ 
for  $l \in \{2, 4, 8, \dots, 2^{\lfloor \ln L \rfloor}\}$  do
    for  $v \in \mathcal{S}, t \in \{0, 1, \dots, L - 1 - l\}$  in parallel do
         $\text{End}(v, t, l) \leftarrow \text{End}(\text{End}(v, t, \frac{l}{2}), t, \frac{l}{2})$ 
for  $l \in 1, 2, \dots, L$  in parallel do
    Decompose  $l$  as a sum of subset  $S \subset \{1, 2, \dots, 2^{\lfloor \ln L \rfloor - 1}\}$ 
     $e \leftarrow (*, v_0)$ 
     $t \leftarrow 0$ 
    for  $s \in S$  do
         $e \leftarrow \text{End}(e, t, s)$ 
         $t \leftarrow t + s$ 
     $e_l \leftarrow e$ 
     $v_l \leftarrow$  the ending vertex of  $e_l$ 
if  $e_k = e_{\text{end}}$  for some  $k$  then
     $\text{return } \mathcal{S}_{v_0}, e_1, \mathcal{S}_{v_1}, e_2, \dots, \mathcal{S}_{v_{k-1}}, e_k = e_{\text{end}}$ 
else
     $\text{return } \mathcal{S}_{v_0}, e_1, \mathcal{S}_{v_1}, e_2, \dots, \mathcal{S}_{v_{L-1}}, e_L, \mathcal{S}, e_{\text{end}}$ 

```

---

► **Remark 18.** For clarity of exposition, we allow sampling numbers from  $[0, 1]$ . However sampling a number with polynomially many bits is enough for our purposes. In our algorithms, we only need to compare our samples  $X_t$  with deterministic numbers derived from the input. To this end, we can simply sample the first  $N$  digits of the binary expansion of  $X_t$  for some large  $N$ . The probability that the comparison of  $X_t$  and a fixed number is not determined from the first  $N$  digits is  $2^{-N}$ . By taking  $N$  to be polynomially large, the probability of failure will be bounded by an exponentially small number. If one insists on avoiding even this small probability of failure, we can continue sampling digits of  $X_t$  whenever we run into a situation where the first  $N$  are not enough. Since the probability of running into failure is very small, the overall expected running time still remains polylogarithmic.

## 5.2 Extracting First-Visit Edges

For a cluster, once we know which edges jump across its children, we can recursively fill in the transcript of the walk. This is because we now have a similar subproblem for each child node, where we have a starting vertex, and a prespecified exit edge. However, a naive implementation of this is not in RNC.

Instead our strategy is to again use a doubling trick. Consider a node  $\mathcal{S}$  of  $\mathcal{T}$  and a vertex  $v \in \mathcal{S}$ . Using the algorithm **JumpingEdges**, we can extract the edges that jump across children of  $\mathcal{S}$ . We can then extend these to include edges that jump across children of children of  $\mathcal{S}$  by more applications of **JumpingEdges**. We will construct a function **AllEdges** $(\mathcal{S}, v, e, L, l)$  that besides the arguments to **JumpingEdges** also takes an integer  $l \geq 1$ . The goal of this function is to extract from the transcript of a random walk started at  $v$  and conditioned on exiting  $\mathcal{S}$  through  $e$ , the first  $L$  edges which jump between descendants of depth at most  $l$  below  $\mathcal{S}$  in  $\mathcal{T}$ . Then, each intermediate node in the transcript is either an individual vertex, an  $l$ th descendant of  $\mathcal{S}$ , or a cluster for which  $L$  edges jumping between its children have already been found. We will show that for polynomially large  $L$ , this transcript will contain all first-visit edges with high probability.

A natural approach for computing **AllEdges** is as follows. We already know how to compute **AllEdges** for  $l = 1$ ; that is just a call to **JumpingEdges**. Once the value of **AllEdges** has been computed for all settings of parameters for a particular  $l$ , we can compute it for  $2l$  by the following procedure:

1. Let the transcript returned by **AllEdges** $(\mathcal{S}, v, e, L, l)$  be  $\mathcal{S}_1, e_1, \mathcal{S}_2, e_2, \dots, \mathcal{S}_k, e_k$ .
2. For each intermediate node  $\mathcal{S}_i$  that is not a single vertex, if the transcript does not already contain  $L$  edges jumping between children of  $\mathcal{S}_i$ , let  $u$  be the endpoint of  $e_{i-1}$  and replace  $\mathcal{S}_i, e_i$  with **AllEdges** $(\mathcal{S}_i, u, e, L, l)$
3. Trim the transcript to only contain at most the first  $L$  edges jumping between children of any node

There is one key problem with this approach. We cannot precompute **AllEdges** $(\cdot, \cdot, \cdot, \cdot, l)$  and use the same precomputed answer to fill in the gaps for larger values of  $l$ . Each time we need to use **AllEdges** for a particular setting of its input parameters, we really need to use fresh randomness; otherwise the transcript we extract will not be from a true random walk.

We resolve this by using a caching trick. Instead of computing **AllEdges** $(\mathcal{S}, v, e, L, l)$  for each setting of parameters once and reusing the same output for larger subproblems, we compute  $M$  possible answers for a large enough  $M$  and store all  $M$  answers. For larger subproblems, every time that we need to use the values of a smaller subproblem, we randomly pick one of the stored  $M$  answers. We will inevitably reuse some of the answers in this process; however when we restrict our attention to the unraveling of a particular answer for a subproblem there is a high probability of not having reused any answers. We will formalize this in Lemma 19.

Pseudocode for computing **AllEdges** can be found in Algorithm 5. In the end we use  $\text{AllEdges}(V, v, \emptyset, L, |V|)$  to extract a list of edges, and with high probability all first-visit edges will be among this list

■ **Algorithm 5** Computing the random walk transcript extracts.

---

```

for  $\mathcal{S} \in \mathcal{T}, e \in \delta^-(\mathcal{S}) \cup \{\emptyset\}, v \in \mathcal{S}, i \in [M]$  in parallel do
   $\text{AllEdges}(\mathcal{S}, v, e, L, 1)[i] \leftarrow \text{JumpingEdges}(\mathcal{S}, v, e, L)$ 
for  $l \in \{1, 2, \dots, 2^{\lceil \lg L \rceil - 1}\}$  do
  for  $\mathcal{S} \in \mathcal{T}, e \in \delta^-(\mathcal{S}) \cup \{\emptyset\}, v \in \mathcal{S}, i \in [M]$  in parallel do
    if  $l$  is greater than depth of the deepest child of  $\mathcal{S}$  then
       $\text{AllEdges}(\mathcal{S}, v, e, L, 2l)[i] \leftarrow \text{AllEdges}(\mathcal{S}, v, e, L, l)[i]$ 
    else
      Let  $\mathcal{S}_1, e_1, \mathcal{S}_2, e_2, \dots, \mathcal{S}_k$  be the output of  $\text{AllEdges}(\mathcal{S}, v, e, L, l)[i]$ 
      for  $j \in [k]$  in parallel do
        if the transcript does not already contain  $L$  edges jumping between
          children of  $\mathcal{S}_j$  then
          Replace  $\mathcal{S}_j, e_j$  with a randomly chosen instance of
             $\text{AllEdges}(\mathcal{S}_j, v', e_j, L, l)$  where  $v'$  is the endpoint of  $e_{j-1}$ 
      Trim the resulting transcript to only contain at most the first  $L$  edges
        jumping between children of any node. Save the resulting transcript as a
        solution to  $\text{AllEdges}(\mathcal{S}, v, e, L, 2l)[i]$ 
  if the final transcript of  $\text{AllEdges}(V, v, \emptyset, L, |V|)$  contain multiple subsequences
    which depend on the same call to JumpingEdges then
      Replace all but the first subsequence with a freshly sampled transcript

```

---

► **Lemma 19.** *For  $M = \text{poly}(L)$ , when we unravel the recursion tree for the computation of the value  $\text{AllEdges}(V, v, \emptyset, L, |V|)$ , no stored answer of **AllEdges** for any of the subproblems will be used more than once with high probability.*

**Proof.** Note that for any  $\mathcal{S} \in \mathcal{T}, v \in \mathcal{S}, e \in \delta^-(\mathcal{S})$ , the recursion tree for an answer to  $\text{AllEdges}(V, v, \emptyset, L, |V|)$  will rely on at most  $L$  calls to  $\text{JumpingEdges}(\mathcal{S}, v, e, L)$  since each call to  $\text{JumpingEdges}(\mathcal{S}, v, e, L)$  corresponds to a subsequence of the final transcript which ends in  $e$ , and  $e$  can only occur  $L$  times in the transcript by the definition of **AllEdges**. As each sample of  $\text{JumpingEdges}(\mathcal{S}, v, e, L)$  is chosen uniformly at random from  $M$  samples, for polynomially large  $M$ , the probability of the same sample being chosen twice can be made small and of the form  $\frac{1}{\text{poly}(n)}$ . Lastly via a union bound over polynomially many possible combinations of arguments for **JumpingEdges**, we have that for polynomially large  $M$ , which high probability no stored answer of **JumpingEdges** or **AllEdges** will be used more than once. ◀

Putting everything together, we have shown that:

► **Theorem 20.** *Suppose that  $G = (V, E)$  is a directed graph, with weights given by  $w : E \rightarrow \mathbb{R}_{>0}$ . Let  $\mathcal{T}$  denote a hierarchical decomposition of  $G$ . The transcript of a random walk starting at  $v_0 \in V$  which contains first  $L$  edges jumping between children of  $\mathcal{S}$  for any  $\mathcal{S} \in \mathcal{T}$  can be produced in the PRAM model using  $\text{poly}(|E|, L)$  processors in  $\text{polylog}(|E|, L)$  expected time.*

## 6 Hierarchical Exploration Time

The last element needed to obtain an RNC algorithm for the random sampling of arborescences is to show that for the decomposition achieved above, for polynomially large  $L$ , simulating the first  $L$  edges which jump across each node  $S \in \mathcal{T}$  will contain all first-visit edges to each vertex with high probability. We do this by bounding the number of edge traversals between children of each node before that node is covered. We build on techniques developed by Boczkowski, Peres, and Sousi [5].

We start by bounding the number of times a given vertex is visited before cover time. For any  $v, s, t \in V$ , let  $H_v(s, t)$  denote the expected number of times a random walk starting at  $s$  visits  $v$  before reaching  $t$ ; in the case that  $s = t$ ,  $H_v(s, s)$  denotes the number of times a walk starting at  $s$  reaches  $v$  before returning to  $s$ .

It is easy to see that  $H_v$  satisfies a triangle inequality, namely:

► **Proposition 21.** *On any directed graph  $G = (V, E)$ , for any  $v, s, t, u \in V$ :  $H_v(s, t) \leq H_v(s, u) + H_v(u, t)$ .*

Additionally, as the stationary distribution of vertices on an Eulerian graph is proportional to the degree of a vertex, it follows that:

► **Proposition 22.** *On any directed Eulerian graph  $G = (V, E)$ , for any  $v, s \in V$ :  $H_v(s, s) = \frac{1}{\pi(s)} \pi(v) = \frac{\deg(v)}{\deg(s)}$*

**Proof.** Consider the Schur complement  $G_S$  where  $S = \{v, s\}$ . Since  $G$  is Eulerian, so is  $G_S$  and the degrees of  $v, s$  are the same in  $G$  and  $G_S$ . Letting the weights of edges in  $G_S$  be  $w(v, s)$ ,  $w(s, v)$ ,  $w(v, v)$ ,  $w(s, s)$ , we must have  $w(s, v) = w(v, s)$  for  $G_S$  to be Eulerian. As random walks on  $G_S$  are distributed like random walks on  $G$  whose transcripts have been restricted to only contain vertices in  $S$ , the values of  $H_v(s, s)$ ,  $H_v(v, s)$  are the same for a walk on  $G$  as for one on  $G_S$ . Considering one-step transitions, we can construct the following system of equations:

$$\begin{aligned} H_v(s, s) &= \frac{w(s, v)}{\deg(s)} (H_v(v, s) + 1) + \frac{w(s, s)}{\deg(s)} 0, \\ H_v(v, s) &= \frac{w(v, v)}{\deg(v)} (H_v(v, s) + 1) + \frac{w(v, s)}{\deg(v)} 0. \end{aligned}$$

Solving yields  $H_v(v, s) = \frac{w(v, v)}{w(v, s)}$  and  $H_v(s, s) = \frac{w(s, v)}{\deg(s)} \left( \frac{w(v, v)}{w(v, s)} + 1 \right) = \frac{w(s, v)}{\deg(s)} \frac{\deg(v)}{w(v, s)} = \frac{\deg(v)}{\deg(s)}$ . ◀

For any two vertices which are connected by an edge, we have:

► **Lemma 23.** *On any directed Eulerian graph  $G = (V, E)$ , for any  $v, s, t \in V$  such that  $e = (s, t) \in E$ ,  $H_v(s, t) \leq \frac{\deg(v)}{w(s, t)}$*

**Proof.** Note that every time the walk is currently at  $s$ , there is a  $\frac{w(s, t)}{\deg(s)}$  chance of moving to  $t$ , thus, in expectation, the expected number of times the walk returns to  $s$  before reaching  $t$  is at most  $\frac{\deg(s)}{w(s, t)}$ . Between each return to  $s$ , the expected number of visits to  $v$  is  $\frac{\deg(v)}{\deg(s)}$  by Proposition 22. Thus, we conclude that:

$$H_v(s, t) \leq \frac{\deg(s)}{w(s, t)} \frac{\deg(v)}{\deg(s)} = \frac{\deg(v)}{w(s, t)}. \quad \blacktriangleleft$$

On graphs which are strongly connected by edges of some weight  $w_c$ , this observation leads to the conclusion that:

► **Lemma 24.** *On an Eulerian directed graph  $G = (V, E)$  strongly connected by edges of weight  $w_c$ , for any  $s, t \in V$ ,  $H_s(s, t) \leq \frac{\deg(s)n}{w_c}$*

**Proof.** By connectivity assumptions, there exists a sequence of at most  $n$  vertices  $s = v_0, v_1, \dots, v_k = t$  which form a path connecting  $s$  and  $t$  such that  $w(v_{i-1}, v_i) \geq w_c$  for all  $i \in [k]$ . Then, combining Proposition 21 with Lemma 23,

$$H_s(s, t) \leq \sum_{i=1}^k H_s(v_{i-1}, v_i) \leq \sum_{i=1}^k \frac{\deg(s)}{w(v_{i-1}, v_i)} \leq \frac{n \deg(s)}{w_c}. \quad \blacktriangleleft$$

This trivially bounds the expected number of returns to a vertex  $s$  before a graph  $G$  is covered by a random walk by  $\frac{n^2 \deg(s)}{w_c}$ .

► **Remark 25.** While this bound is sufficient for our purposes, using Matthew's trick, this bound can be further tightened to  $\frac{n \log n \deg(s)}{w_c}$  [26].

► **Lemma 26.** *For any node  $\mathcal{S} \in \mathcal{T}$  in the decomposition obtained by Algorithm 2, the expected number of edges traversed between children of  $\mathcal{S}$  before every vertex in  $\mathcal{S}$  has been visited is at most  $n^2 m^2$ .*

**Proof.** For each edge  $e = (u, v)$  which jumps between children of  $\mathcal{S}$ , the expected number of times edge  $e$  is traversed is  $\frac{w_e}{\deg(u)}$  times the expected number of times vertex  $u$  is reached by a random walk before all vertices in  $\mathcal{S}$  are reached. To bound the number of times  $u$  is reached before  $\mathcal{S}$  is covered, consider a random walk on the Schur complement  $G_{\mathcal{S}}$ . The expected number of times  $u$  is reached in a random walk on  $G$  before  $\mathcal{S}$  is covered is the same as the expected number of times  $u$  is reached by a random walk on  $G_{\mathcal{S}}$  before  $G_{\mathcal{S}}$  is covered.

By Proposition 11,  $G_{\mathcal{S}}$  will be Eulerian and the degrees of all vertices in  $\mathcal{S}$  will be the same in  $G$  and  $G_{\mathcal{S}}$ . By Lemma 16,  $\mathcal{S}$  is strongly connected by edges of weight  $\frac{w_{\max}(\mathcal{S})}{m}$ , and so  $G_{\mathcal{S}}$  is as well. Then, by Lemma 24, the expected number of times  $u$  is hit before  $G_{\mathcal{S}}$  is covered is at most by  $\frac{\deg(u)n^2 m}{w_{\max}}$ . Thus, the expected number of times edge  $e$  is traversed before  $\mathcal{S}$  is covered is at most

$$\frac{\deg(u)n^2 m}{w_{\max}} \frac{w_e}{\deg(u)} \leq n^2 m$$

since  $w_c \geq \frac{w_e}{m}$ . As there are at most  $m$  jumping edges of  $\mathcal{S}$ , in expectation, at most  $n^2 m^2$  edge traversals between children of  $\mathcal{S}$  will occur before every vertex in  $\mathcal{S}$  has been reached.  $\blacktriangleleft$

Thus, for large enough  $L = \text{poly}(n, m)$ , by Markov's inequality, with high probability every node  $\mathcal{S} \in \mathcal{T}$  is covered by the time  $L$  edges have jumped across  $\mathcal{S}$ . This means the transcript returned by `AllEdges` will contain all first visit edges with high probability. As this transcript is polynomial in length, all first visit edges and the corresponding arborescence they form can be extracted and returned in NC.

## 7 Discussion and Open Problems

We showed how to sample arborescences, and as a special case spanning trees, from a given weighted graph using an RNC algorithm. While this is a step in resolving the disparity between parallel sampling and parallel counting algorithms, more investigation is needed. In particular, for the list of problems with determinant-based counting in Section 1, designing RNC sampling algorithms remains open.

One of these problems in particular, namely sampling Eulerian tours from an Eulerian digraph, is intimately connected to sampling arborescences, due to the BEST theorem [1, 34]. However the reduction, while polynomial-time implementable, is not known to be in NC.

► **Question 27.** Is there an algorithm for sampling Eulerian tours uniformly at random in Eulerian digraphs?

The important tool our result relied on was the Aldous-Broder algorithm. So it is natural to ask whether there are generalizations of the Aldous-Broder result to settings beyond spanning trees and arborescences. In particular the bases of a *regular matroid* are a proper generalization of spanning trees in a graph, and they have a decomposition in terms of graphic, co-graphic, and some special constant-sized matroids [29].

► **Question 28.** Can we sample from a regular matroid, or equivalently from a volume-based distribution defined by totally unimodular vectors in RNC?

Yet another direction for generalization are higher-dimensional equivalents of spanning trees and arborescences. For example, Gorodezky and Pak [15] provided a generalization of the algorithm of Wilson [35] for sampling arborescences from graphs to hypergraphs. While Wilson’s algorithm is different from that of Aldous-Broder, it is closely related. Can these generalizations be efficiently parallelized?

---

## References

- 1 T Aardenne-Ehrenfest and NG Bruijn. Circuits and trees in oriented linear graphs. *Classic Papers in Combinatorics*, pages 149–163, 1987.
- 2 David J Aldous. The random walk construction of uniform spanning trees and uniform labelled trees. *SIAM Journal on Discrete Mathematics*, 3.4:450–465, 1990.
- 3 Nima Anari, Kuikui Liu, Shayan Oveis Gharan, and Cynthia Vintant. Log-concave polynomials iv: Exchange properties, tight mixing times, and faster sampling of spanning trees. *arXiv preprint arXiv:2004.07220*, 2020.
- 4 Nikhil Balaji and Samir Datta. Tree-width and logspace: Determinants and counting euler tours. *arXiv preprint*, 2013. [arXiv:1312.7468](#).
- 5 Lucas Boczkowski, Yuval Peres, and Perla Sousi. Sensitivity of mixing times in eulerian digraphs. *SIAM Journal on Discrete Mathematics*, 32(1):624–655, 2018.
- 6 Andrei Broder. Generating random spanning trees. In *Foundations of Computer Science, 1989., 30th Annual Symposium on*, pages 442–447. IEEE, 1989.
- 7 Charles J Colbourn, Wendy J Myrvold, and Eugene Neufeld. Two algorithms for unranking arborescences. *Journal of Algorithms*, 20(2):268–281, 1996.
- 8 Patrick John Creed. *Counting and sampling problems on Eulerian graphs*. PhD thesis, The University of Edinburgh, 2010.
- 9 Laszlo Csanky. Fast parallel matrix inversion algorithms. In *Foundations of Computer Science, 1975., 16th Annual Symposium on*, pages 11–12. IEEE, 1975.
- 10 Amit Deshpande and Luis Rademacher. Efficient volume sampling for row/column subset selection. In *Foundations of Computer Science (FOCS), 2010 51st Annual IEEE Symposium on*, pages 329–338. IEEE, 2010.
- 11 David Durfee, Rasmus Kyng, John Peebles, Anup B Rao, and Sushant Sachdeva. Sampling random spanning trees faster than matrix multiplication. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 730–742, 2017.
- 12 David Durfee, John Peebles, Richard Peng, and Anup B Rao. Determinant-preserving sparsification of sddm matrices with applications to counting and sampling spanning trees. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 926–937. IEEE, 2017.
- 13 Martin Dyer, Alan Frieze, and Ravi Kannan. A random polynomial-time algorithm for approximating the volume of convex bodies. *Journal of the ACM (JACM)*, 38(1):1–17, 1991.
- 14 Ira M Gessel and Xavier Viennot. Determinants, paths, and plane partitions. *preprint*, 132(197.15), 1989.



- 15 Igor Gorodezky and Igor Pak. Generalized loop-erased random walks and approximate reachability. *Random Structures & Algorithms*, 44(2):201–223, 2014.
- 16 Mark Jerrum and Alistair Sinclair. The markov chain monte carlo method: an approach to approximate counting and integration. *Approximation algorithms for NP-hard problems*, pages 482–520, 1996.
- 17 Mark Jerrum, Alistair Sinclair, and Eric Vigoda. A polynomial-time approximation algorithm for the permanent of a matrix with nonnegative entries. *Journal of the ACM (JACM)*, 51(4):671–697, 2004.
- 18 Mark R Jerrum, Leslie G Valiant, and Vijay V Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theoretical computer science*, 43:169–188, 1986.
- 19 Minghui Jiang, James Anderson, Joel Gillespie, and Martin Mayne. ushuffle: a useful tool for shuffling biological sequences while preserving the k-let counts. *BMC bioinformatics*, 9(1):192, 2008.
- 20 Pieter W Kasteleyn. Dimer statistics and phase transitions. *Journal of Mathematical Physics*, 4(2):287–293, 1963.
- 21 Jonathan A Kelner and Aleksander Madry. Faster generation of random spanning trees. In *Foundations of Computer Science, 2009. FOCS’09. 50th Annual IEEE Symposium on*, pages 13–21. IEEE, 2009.
- 22 Gustav Kirchhoff. Ueber die auflösung der gleichungen, auf welche man bei der untersuchung der linearen vertheilung galvanischer ströme geführt wird. *Annalen der Physik*, 148(12):497–508, 1847.
- 23 David A Levin and Yuval Peres. *Markov chains and mixing times*, volume 107. American Mathematical Soc., 2017.
- 24 Bernt Lindström. On the vector representations of induced matroids. *Bulletin of the London Mathematical Society*, 5(1):85–90, 1973.
- 25 Aleksander Mądry, Damian Straszak, and Jakub Tarnawski. Fast generation of random spanning trees and the effective resistance metric. In *Proceedings of the twenty-sixth annual ACM-SIAM symposium on Discrete algorithms*, pages 2019–2036. Society for Industrial and Applied Mathematics, 2015.
- 26 Peter Matthews. Covering problems for brownian motion on spheres. *The Annals of Probability*, 16(1):189–199, 1988. doi:10.1214/aop/1176991894.
- 27 Romain Rivière, Dominique Barth, Johanne Cohen, and Alain Denise. Shuffling biological sequences with motif constraints. *Journal of Discrete Algorithms*, 6(2):192–204, 2008.
- 28 Aaron Schild. An almost-linear time algorithm for uniform random spanning tree generation. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 214–227. ACM, 2018.
- 29 Paul D Seymour. Decomposition of regular matroids. *Journal of combinatorial theory, Series B*, 28(3):305–359, 1980.
- 30 Shang-Hua Teng. Independent sets versus perfect matchings. *Theoretical Computer Science*, 145(1-2):381–390, 1995.
- 31 Shang-Hua Teng. Independent sets versus perfect matchings. *Theoretical Computer Science*, 145.1-2:381–390, 1995.
- 32 Prasad Tetali and Santosh Vempala. Random sampling of euler tours. *Algorithmica*, 30(3):376–385, 2001.
- 33 W. T. Tutte. The dissection of equilateral triangles into equilateral triangles. *Mathematical Proceedings of the Cambridge Philosophical Society*, 44(4):463–482, 1948. doi:10.1017/S030500410002449X.
- 34 WT Tutte and Cedric AB Smith. On unicursal paths in a network of degree 4. *The American Mathematical Monthly*, 48(4):233–237, 1941.
- 35 David Bruce Wilson. Generating random spanning trees more quickly than the cover time. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 296–303, 1996.

# Non-Quasi-Linear Agents in Quasi-Linear Mechanisms

**Moshe Babaioff**

Microsoft Research, Herzliya, Israel  
moshe@microsoft.com

**Richard Cole**

New York University, New York City, NY, USA  
cole@cs.nyu.edu

**Jason Hartline**

Northwestern University, Evanston, IL, USA  
hartline@eecs.northwestern.edu

**Nicole Immorlica**

Microsoft Research, Cambridge, MA, USA  
nicimm@microsoft.com

**Brendan Lucier**

Microsoft Research, Cambridge, MA, USA  
brlucier@microsoft.com

---

## Abstract

Mechanisms with money are commonly designed under the assumption that agents are quasi-linear, meaning they have linear disutility for spending money. We study the implications when agents with non-linear (specifically, convex) disutility for payments participate in mechanisms designed for quasi-linear agents. We first show that any mechanism that is truthful for quasi-linear buyers has a simple best response function for buyers with non-linear disutility from payments, in which each bidder simply scales down her value for each potential outcome by a fixed factor, equal to her target return on investment (ROI). We call such a strategy ROI-optimal. We prove the existence of a Nash equilibrium in which agents use ROI-optimal strategies for a general class of allocation problems. Motivated by online marketplaces, we then focus on simultaneous second-price auctions for additive bidders and show that all ROI-optimal equilibria in this setting achieve constant-factor approximations to suitable welfare and revenue benchmarks.

**2012 ACM Subject Classification** Theory of computation → Quality of equilibria

**Keywords and phrases** Return on investment, Non-quasi-linear agents, Transferable Welfare, Simultaneous Second-Price Auctions

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.84

**Category** Extended Abstract

**Related Version** Full Version: <https://arxiv.org/abs/2012.02893>

**Funding** *Richard Cole*: This work was supported in part by NSF Grant CCF-1909538.

*Jason Hartline*: This work was supported in part by NSF Grant CCF-1618502.



© Moshe Babaioff, Richard Cole, Jason Hartline, Nicole Immorlica, and Brendan Lucier;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 84; pp. 84:1–84:1



Leibniz International Proceedings in Informatics

LIPIcs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



# A Model for Ant Trail Formation and its Convergence Properties

**Moses Charikar**

Stanford University, CA, USA  
moses@cs.stanford.edu

**Shivam Garg**

Stanford University, CA, USA  
shivamgarg@stanford.edu

**Deborah M. Gordon**

Stanford University, CA, USA  
dmgordon@stanford.edu

**Kirankumar Shiragur**

Stanford University, CA, USA  
shiragur@stanford.edu

---

## Abstract

We introduce a model for ant trail formation, building upon previous work on biologically feasible local algorithms that plausibly describe how ants maintain trail networks. The model is a variant of a reinforced random walk on a directed graph, where ants lay pheromone on edges as they traverse them and the next edge to traverse is chosen based on the level of pheromone; this pheromone decays with time. There is a bidirectional flow of ants in the network: the forward flow proceeds along forward edges from source (e.g. the nest) to sink (e.g. a food source), and the backward flow in the opposite direction. Some fraction of ants are lost as they pass through each node (modeling the loss of ants due to exploration observed in the field). We initiate a theoretical study of this model. We note that ant navigation has inspired the field of ant colony optimization, heuristics that have been applied to several combinatorial optimization problems; however the algorithms developed there are considerably more complex and not constrained to being biologically feasible.

We first consider the linear decision rule, where the flow divides itself among the next set of edges in proportion to their pheromone level. Here, we show that the process converges to the path with minimum leakage when the forward and backward flows do not change over time. On the other hand, when the forward and backward flows increase over time (caused by positive reinforcement from the discovery of a food source, for example), we show that the process converges to the shortest path. These results are for graphs consisting of two parallel paths (a case that has been investigated before in experiments). Through simulations, we show that these results hold for more general graphs drawn from various random graph models; proving this convergence in the general case is an interesting open problem. Further, to understand the behaviour of other decision rules beyond the linear rule, we consider a general family of decision rules. For this family, we show that there is no advantage of using a non-linear decision rule, if the goal is to find the shortest or the minimum leakage path. We also show that bidirectional flow is necessary for convergence to such paths. Our results provide a plausible explanation for field observations, and open up new avenues for further theoretical and experimental investigation.

**2012 ACM Subject Classification** Theory of computation → Random walks and Markov chains; Applied computing → Biological networks; Mathematics of computing → Graph algorithms; Theory of computation → Shortest paths

**Keywords and phrases** Ant colonies, Reinforced random walks, Natural Algorithms, Graph Algorithms, Shortest Path, Distributed Algorithms

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.85

**Category** Extended Abstract



© Moses Charikar, Shivam Garg, Deborah M. Gordon, and Kirankumar Shiragur; licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 85; pp. 85:1–85:2



Leibniz International Proceedings in Informatics

LIPIcs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

**Related Version** A full version of the paper is available at [1], <https://arxiv.org/abs/2011.14722>.

**Funding** *Moses Charikar*: Supported by a Simons Investigator Award, a Google Faculty Research Award and an Amazon Research Award.

*Shivam Garg*: Supported by NSF awards AF-1813049 and AF-1704417, and a Stanford Interdisciplinary Graduate Fellowship.

*Deborah M. Gordon*: Supported the Templeton Fund.

*Kirankumar Shiragur*: Supported by a Stanford Data Science Scholarship and a Dantzig-Lieberman Operations Research Fellowship.

**Acknowledgements** We are thankful to the anonymous reviewers for their helpful comments.

---

## References

- 1     Moses Charikar, Shivam Garg, Deborah M. Gordon, and Kirankumar Shiragur. A model for ant trail formation and its convergence properties, 2020. [arXiv:2011.14722](https://arxiv.org/abs/2011.14722).

# Unknown I.I.D. Prophets: Better Bounds, Streaming Algorithms, and a New Impossibility

**José Correa**

Departamento de Ingeniería Industrial, Universidad de Chile, Santiago, Chile  
correa@uchile.cl

**Paul Dütting**

Department of Mathematics, London School of Economics, United Kingdom  
p.d.duetting@lse.ac.uk

**Felix Fischer**

School of Mathematical Sciences, Queen Mary University of London, United Kingdom  
felix.fischer@qmul.ac.uk

**Kevin Schewior**

Department Mathematik/Informatik, Universität zu Köln, Germany  
kschewior@gmail.com

**Bruno Ziliotto**

CEREMADE, CNRS, PSL Research Institute, Université Paris Dauphine, Paris, France  
ziliotto@math.cnrs.fr

---

## Abstract

A prophet inequality states, for some  $\alpha \in [0, 1]$ , that the expected value achievable by a gambler who sequentially observes random variables  $X_1, \dots, X_n$  and selects one of them is at least an  $\alpha$  fraction of the maximum value in the sequence. We obtain three distinct improvements for a setting that was first studied by Correa et al. (EC, 2019) and is particularly relevant to modern applications in algorithmic pricing. In this setting, the random variables are i.i.d. from an unknown distribution and the gambler has access to an additional  $\beta n$  samples for some  $\beta \geq 0$ . We first give improved lower bounds on  $\alpha$  for a wide range of values of  $\beta$ ; specifically,  $\alpha \geq (1 + \beta)/e$  when  $\beta \leq 1/(e - 1)$ , which is tight, and  $\alpha \geq 0.648$  when  $\beta = 1$ , which improves on a bound of around 0.635 due to Correa et al. (SODA, 2020). Adding to their practical appeal, specifically in the context of algorithmic pricing, we then show that the new bounds can be obtained even in a streaming model of computation and thus in situations where the use of relevant data is complicated by the sheer amount of data available. We finally establish that the upper bound of  $1/e$  for the case without samples is robust to additional information about the distribution, and applies also to sequences of i.i.d. random variables whose distribution is itself drawn, according to a known distribution, from a finite set of known candidate distributions. This implies a tight prophet inequality for exchangeable sequences of random variables, answering a question of Hill and Kertz (Contemporary Mathematics, 1992), but leaves open the possibility of better guarantees when the number of candidate distributions is small, a setting we believe is of strong interest to applications.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Online algorithms

**Keywords and phrases** Prophet Inequalities, Stopping Theory, Unknown Distributions

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.86

**Category** Extended Abstract

**Related Version** Full version hosted on arXiv: <https://arxiv.org/abs/2007.06110>.

**Funding** *José Correa*: Supported in part by ANID Chile through grant CMM-AFB 170001.

*Paul Dütting*: Work done in part while author was at Google Research, Zürich, Switzerland.

*Felix Fischer*: Supported in part by EPSRC grant EP/T015187/1.

*Kevin Schewior*: Supported in part by DAAD within the PRIME program.



© José Correa, Paul Dütting, Felix Fischer, Kevin Schewior, and Bruno Ziliotto;  
licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 86; pp. 86:1–86:1



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany





# Complexity Measures on the Symmetric Group and Beyond

**Neta Dafni**

Department of Computer Science, Technion, Haifa, Israel  
netad@cs.technion.ac.il

**Yuval Filmus** 

Department of Computer Science, Technion, Haifa, Israel  
<https://yuvalfilmus.cs.technion.ac.il/>  
yuvalfi@cs.technion.ac.il

**Noam Lifshitz**

Einstein Institute of Mathematics, Hebrew University of Jerusalem, Israel  
noamlifshitz@gmail.com

**Nathan Lindzey**

Department of Computer Science, University of Colorado, Boulder, CO, USA  
<http://nathanlindzey.com/>  
Nathan.Lindzey@colorado.edu

**Marc Vinyals**

Department of Computer Science, Technion, Haifa, Israel  
<http://www.csc.kth.se/~vinyals/>  
marcviny@cs.technion.ac.il

---

## Abstract

We extend the definitions of complexity measures of functions to domains such as the symmetric group. The complexity measures we consider include degree, approximate degree, decision tree complexity, sensitivity, block sensitivity, and a few others. We show that these complexity measures are polynomially related for the symmetric group and for many other domains.

To show that all measures but sensitivity are polynomially related, we generalize classical arguments of Nisan and others. To add sensitivity to the mix, we reduce to Huang’s sensitivity theorem using “pseudo-characters”, which witness the degree of a function.

Using similar ideas, we extend the characterization of Boolean degree 1 functions on the symmetric group due to Ellis, Friedgut and Pilpel to the perfect matching scheme. As another application of our ideas, we simplify the characterization of maximum-size  $t$ -intersecting families in the symmetric group and the perfect matching scheme.

**2012 ACM Subject Classification** Theory of computation → Computational complexity and cryptography; Mathematics of computing → Discrete mathematics

**Keywords and phrases** Computational Complexity Theory, Analysis of Boolean Functions, Complexity Measures, Extremal Combinatorics

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.87

**Category** Extended Abstract

**Related Version** A full version of the paper is available at <https://arxiv.org/abs/2010.07405>.

**Funding** This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 802020-ERC-HARMONIC.

**Acknowledgements** We thank Nitin Saurabh for many helpful discussions.



© Neta Dafni, Yuval Filmus, Noam Lifshitz, Nathan Lindzey, and Marc Vinyals;  
licensed under Creative Commons License CC-BY  
12th Innovations in Theoretical Computer Science Conference (ITCS 2021).  
Editor: James R. Lee; Article No. 87; pp. 87:1–87:5



Leibniz International Proceedings in Informatics  
LIPIcs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

## 1 Introduction

A classical result in complexity theory states that a Boolean function  $f: \{0, 1\}^n \rightarrow \{0, 1\}$  of degree  $d$  can be computed using a decision tree of depth  $\text{poly}(d)$ . Conversely, a Boolean function computed by a decision tree of depth  $d$  has degree at most  $d$ . Thus degree and decision tree complexity are polynomially related. Other complexity measures which are polynomially related to the degree include approximate degree, certificate complexity, and block sensitivity. Recently, Huang [9] added sensitivity to the list.

Can we prove similar results for Boolean functions on other domains? Such domains have been introduced to complexity theory in recent years: for example, O'Donnell and Wimmer [13] used Boolean functions on the so-called “slice” to construct optimal nets for monotone functions; Barak et al. [1] used Boolean functions on the Reed–Muller code to construct and analyze the influential “short code”; and recently, Khot, Minzer and Safra [10] proved the 2-to-2 conjecture using Boolean functions on the Grassmann scheme.

Although yet to see applications to complexity theory, perhaps the most appealing domain is the symmetric group. We say that a function  $f: S_n \rightarrow \mathbb{R}$  has degree at most  $d$  if any of the following equivalent conditions hold:

1.  $f(\pi)$  can be written as a linear combination of  $d$ -juntas, which are functions depending on  $\pi(i_1), \dots, \pi(i_d)$  for some  $i_1, \dots, i_d \in [n]$ .
2. Representing the input as a permutation matrix,  $f$  can be written as a degree  $d$  polynomial in the entries of the matrix.
3.  $f$  has Fourier-degree  $d$ , that is, it is supported on isotypic components corresponding to partitions  $\lambda$  with  $\lambda_1 \geq n - d$ .

(The reader who is not familiar with representation theory can ignore the last definition.)

What is the correct generalization of decision tree? We take our inspiration from the work of Ellis, Friedgut and Pilpel [4], which characterized the Boolean degree 1 functions on  $S_n$ . These are functions that depend on some  $\pi(i)$  or on some  $\pi^{-1}(j)$ . This suggests the following definition: a decision tree for functions on  $S_n$  is a decision tree with queries of the form “ $\pi(i) = ?$ ” and “ $\pi^{-1}(j) = ?$ ”. Essentially the same definition (“matching decision trees”) is used in lower bounds on the pigeonhole principle [14].

We show that this is a good definition by proving that degree and decision tree complexity are polynomially related for the symmetric group. In fact, we are able to generalize many other complexity measures to the symmetric group, and show that all of them are polynomially related:

► **Theorem 1.** *The following complexity measures (appropriately defined) are all polynomially related for Boolean functions over the symmetric group: degree, approximate degree, decision tree complexity, certificate complexity, unambiguous certificate complexity, sensitivity, block sensitivity, fractional block sensitivity, quantum query complexity.*

Our results hold for many other domains, such as the perfect matching scheme (the set of all perfect matchings in  $K_{2n}$ ) and balanced slices (the balanced slice consists of all vectors in  $\{0, 1\}^{2n}$  with equally many 0s and 1s, and is also known as the Johnson scheme  $J(2n, n)$ ).

We prove Theorem 1 and its generalizations in an abstract framework based on simplicial complexes. In this framework, every point in the domain is a set. For example:

1. Boolean cube: We identify each vector  $x \in \{0, 1\}^n$  with the set  $\{(i, x_i) : i \in [n]\}$ .
2. Symmetric group: We identify each permutation  $\pi \in S_n$  with the set  $\{(i, \pi(i)) : i \in [n]\}$ .

A function has degree  $d$  if it can be written as a linear combination of functions of the form “the input set contains  $S$ ”, where  $|S| \leq d$ ; this generalizes the usual notion of degree in these two domains. Our decision trees allow any queries of the form “which element of the set  $Q$  does the input set contain?”, as long as there is a unique answer for every input.

With this setup in place, we are able to polynomially relate all complexity measures other than sensitivity by generalizing classical arguments, as presented by Buhrman and de Wolf [2], for example. To add sensitivity to the mix, we reduce to Huang’s sensitivity theorem [9] using basic representation theory.

Generalizing ideas of Gopalan et al. [8], we also prove the following simple result:

► **Theorem 2.** *If a function on the symmetric group has sensitivity  $s$ , then it can be recovered from its evaluation on a ball of radius  $O(s)$  around an arbitrary permutation.*

Using this, we show that low sensitivity functions can be computed efficiently:

► **Theorem 3.** *If a function on the symmetric group has sensitivity  $s$ , then it can be computed using a circuit of size  $n^{O(s)}$ .*

This should be compared to a decision tree for the function, which corresponds to a balanced formula of size  $n^{O(D)}$ , where  $D \geq s$  is the decision tree complexity.

## 1.1 Degree 1 functions

Our results show that in a wide variety of domains, Boolean degree 1 functions can be computed by constant depth decision trees. Can we say more?

Boolean degree 1 functions on the Boolean cube are *dictators*, that is, depend on a single coordinate, and the same holds for functions on the balanced slice. Ellis, Friedgut and Pilpel [4] showed that the same holds for the symmetric group, with the correct interpretation of “dictator”: a function depending only on some  $\pi(i)$ , or only on some  $\pi^{-1}(j)$ . In contrast, Filmus and Ihringer [6] showed that Boolean degree 1 functions on the Grassmann scheme ( $k$ -dimensional subspaces of an  $n$ -dimensional vector space over a finite field) could depend on two different “data points”.

Among the domains we consider, in many cases Boolean degree 1 functions are trivially dictators. In some other cases, describing all Boolean degree 1 functions seems difficult. We identify one case in which the problem is feasible:

► **Theorem 4.** *A Boolean function on the perfect matching scheme has degree at most 1 if and only if it is one of the following: a constant function; a function depending on the match of some vertex  $i$ ; or a function depending on whether the perfect matching intersects some triangle.*

Incidentally, this is another example in which there are non-dictatorial degree 1 functions, namely those depending on intersections with a triangle.

We prove Theorem 4 using polyhedral techniques. As in the proof of the corresponding result for the symmetric group by Ellis, Friedgut and Pilpel (which we paraphrase using our methods), we first characterize all *nonnegative* degree 1 functions, using the classical characterization of supporting hyperplanes of the perfect matching polytope. To deduce the result for Boolean functions, we use a simple result from the theory of complexity measures: a degree 1 function has sensitivity at most 1.

The reader is perhaps wondering about nonnegative functions of higher degree. Can we say anything intelligent about them? It turns out that the answer is negative already for the Boolean cube: classical results on the Sherali–Adams hierarchy [7] show that there exist nonnegative degree 2 functions which, if written as nonnegative linear combinations of monomials over literals (that is, products of factors of the form  $x_i$  and  $1 - x_j$ ), require degree  $\Omega(n)$ .

## 1.2 Application to Erdős–Ko–Rado theory

The work of Ellis, Friedgut and Pilpel, which has already been mentioned several times, is about intersecting families of permutations. A subset  $\mathcal{F} \subseteq S_n$  is *t-intersecting* if any two  $\pi_1, \pi_2 \in \mathcal{F}$  agree on the image of at least  $t$  points. In other words, if we think of  $\pi_1, \pi_2$  as sets (as in our setup), then  $|\pi_1 \cap \pi_2| \geq t$ . How large can a  $t$ -intersecting family be? One construction is a *t-star*:

$$\mathcal{F} = \{\pi \in S_n : \pi(i_1) = j_1, \dots, \pi(i_t) = j_t\}.$$

Ellis et al. show that for large enough  $n$  (depending on  $t$ ), these families have the maximum possible size, and moreover uniquely so: every  $t$ -intersecting family of the maximum size  $(n - t)!$  is a  $t$ -star. Unfortunately, their argument for the uniqueness claim is wrong, see [5]. Uniqueness can be recovered from the work of Ellis [3], which proves a much stronger result, and is quite complicated.

We give a much simpler proof of uniqueness, using the connection between degree and certificate complexity:

► **Theorem 5.** *For every  $t, d$ , the following holds for large enough  $n$ . If  $f$  is the characteristic vector of a  $t$ -intersecting family and  $\deg f \leq d$ , then either  $f$  is contained in a  $t$ -star, or the corresponding family contains  $O((n - t - 1)!)$  permutations.*

Ellis et al. show that a  $t$ -intersecting family of size  $(n - t)!$  must have degree  $t$  (for large enough  $n$ ), and so Theorem 5 shows that for large enough  $n$ , such a family must be a  $t$ -star.

Theorem 5 generalizes to other domains for which similar intersection theorems are known, such as the perfect matching scheme [11, 12], and to cross- $t$ -intersecting families.

We see Theorem 5 as a contribution of theoretical computer science to extremal combinatorics. It illustrates the usefulness of the theory developed in this work.

---

## References

- 1 Boaz Barak, Parikshit Gopalan, Johan Håstad, Raghu Meka, Prasad Raghavendra, and David Steurer. Making the long code shorter. *SIAM J. Comput.*, 44(5):1287–1324, 2015. doi:10.1137/130929394.
- 2 Harry Buhrman and Ronald de Wolf. Complexity measures and decision tree complexity: a survey. *Theoret. Comput. Sci.*, 288(1):21–43, 2002. Complexity and logic (Vienna, 1998). doi:10.1016/S0304-3975(01)00144-X.
- 3 David Ellis. Stability for  $t$ -intersecting families of permutations. *J. Combin. Theory Ser. A*, 118(1):208–227, 2011. doi:10.1016/j.jcta.2010.04.005.
- 4 David Ellis, Ehud Friedgut, and Haran Pilpel. Intersecting families of permutations. *J. Amer. Math. Soc.*, 24(3):649–682, 2011. doi:10.1090/S0894-0347-2011-00690-5.
- 5 Yuval Filmus. A comment on intersecting families of permutations. *arXiv*, abs/1706.10146, 2017. arXiv:1706.10146.
- 6 Yuval Filmus and Ferdinand Ihringer. Boolean degree 1 functions on some classical association schemes. *J. Combin. Theory Ser. A*, 162:241–270, 2019. doi:10.1016/j.jcta.2018.11.006.
- 7 Konstantinos Georgiou, Avner Magen, and Madhur Tulsiani. Optimal Sherali-Adams gaps from pairwise independence. In *Approximation, randomization, and combinatorial optimization*, volume 5687 of *Lecture Notes in Comput. Sci.*, pages 125–139. Springer, Berlin, 2009. doi:10.1007/978-3-642-03685-9\_10.
- 8 Parikshit Gopalan, Noam Nisan, Rocco A. Servedio, Kunal Talwar, and Avi Wigderson. Smooth Boolean functions are easy: efficient algorithms for low-sensitivity functions. In *ITCS'16—Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science*, pages 59–70. ACM, New York, 2016. doi:10.1145/2840728.2840738.

- 9 Hao Huang. Induced subgraphs of hypercubes and a proof of the sensitivity conjecture. *Ann. of Math. (2)*, 190(3):949–955, 2019. doi:10.4007/annals.2019.190.3.6.
- 10 Subhash Khot, Dor Minzer, and Muli Safra. Pseudorandom sets in Grassmann graph have near-perfect expansion. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 592–601, 2018.
- 11 Nathan Lindzey. Intersecting families of perfect matchings. *ArXiv*, abs/1811.06160, 2018.
- 12 Nathan Lindzey. *Matchings and representation theory*. PhD thesis, University of Waterloo, 2018.
- 13 Ryan O’Donnell and Karl Wimmer. KKL, Kruskal-Katona, and monotone nets. *SIAM J. Comput.*, 42(6):2375–2399, 2013. doi:10.1137/100787325.
- 14 Alasdair Urquhart and Xudong Fu. Simplified lower bounds for propositional proofs. *Notre Dame J. Formal Logic*, 37(4):523–544, 1996. doi:10.1305/ndjfl/1040046140.



# Batching and Optimal Multi-Stage Bipartite Allocations

Yiding Feng

Department of Computer Science, Northwestern University, Evanston, IL, USA  
yidingfeng2021@u.northwestern.edu

Rad Niazadeh

Chicago Booth School of Business, University of Chicago, Chicago, IL, USA  
rad.niazadeh@chicagobooth.edu

---

## Abstract

In several applications of real-time matching of demand to supply in online marketplaces, the platform can allow for some latency to batch the demand and improve the matching's efficiency. Motivated by these scenarios, we study the optimal trade-off between batching and inefficiency in online allocations. In particular, we consider  $K$ -stage variants of the classic vertex weighted bipartite b-matching and AdWords problems, where online vertices arrive in  $K$  batches. Our main result for both problems is an optimal  $(1 - (1 - 1/K)^K)$ -competitive (fractional) matching algorithm, improving the classic  $(1 - 1/e)$  competitive ratios known for the online variant of these problems [2, 1].

Our main technique is using a family of convex-programming based matchings that distribute the demand in a particularly balanced way among supply in different stages. More precisely, we identify a sequence of *polynomials with decreasing degrees* that can be used as strictly concave regularizers of the optimal matching linear program to form this family. By providing structural decompositions of the underlying graph using the optimal solutions of these convex programs, we develop a new multi-stage primal-dual framework to analyze the fractional multi-stage algorithm that returns the corresponding regularized optimal matching in each stage (by solving the stage's convex program). We further show a matching upper-bound by providing an unweighted instance of the problem in which no online algorithm obtains a competitive ratio better than  $(1 - (1 - 1/K)^K)$ . We extend our results to integral allocations in the vertex weighted b-matching problem with large budgets, and in the AdWords problem with small bid over budget ratios.

**2012 ACM Subject Classification** Theory of computation → Algorithmic game theory and mechanism design; Theory of computation → Online algorithms

**Keywords and phrases** Online Bipartite Matching, Primal-Dual Analysis, Multi-stage Allocation, Batching

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.88

**Category** Extended Abstract

**Related Version** A full version of the paper is available at <https://ssrn.com/abstract=3689448>.

**Acknowledgements** The authors would like to thank Yeganeh Alimohammadi for participating in the early stages of this project, and to Amin Saberi for helpful discussions regarding the presentation of this work.

---

## References

- 1 Gagan Aggarwal, Gagan Goel, Chinmay Karande, and Aranyak Mehta. Online vertex-weighted bipartite matching and single-bid budgeted allocations. In *Proceedings of the twenty-second annual ACM-SIAM symposium on Discrete Algorithms*, pages 1253–1264. SIAM, 2011.
- 2 Aranyak Mehta, Amin Saberi, Umesh Vazirani, and Vijay Vazirani. Adwords and generalized online matching. *Journal of the ACM (JACM)*, 54(5):22-es, 2007.



© Yiding Feng and Rad Niazadeh;  
licensed under Creative Commons License CC-BY  
12th Innovations in Theoretical Computer Science Conference (ITCS 2021).  
Editor: James R. Lee; Article No. 88; pp. 88:1–88:1




Leibniz International Proceedings in Informatics  
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany





# Shrinkage Under Random Projections, and Cubic Formula Lower Bounds for $\mathbf{AC}^0$

Yuval Filmus 

Technion – Israel Institute of Technology, Haifa, Israel

<https://yuvalfilmus.cs.technion.ac.il>


[yuvalfi@cs.technion.ac.il](mailto:yuvalfi@cs.technion.ac.il)

Or Meir 

Department of Computer Science, University of Haifa, Israel

<https://cs.haifa.ac.il/~ormeir/>

[ormeir@cs.haifa.ac.il](mailto:ormeir@cs.haifa.ac.il)

Avishay Tal 

Department of Electrical Engineering and Computer Sciences,

University of California at Berkeley, CA, USA

<http://www.avishaytal.org>

[atal@berkeley.edu](mailto:atal@berkeley.edu)

---

## Abstract

Håstad showed that any De Morgan formula (composed of AND, OR and NOT gates) shrinks by a factor of  $O(p^2)$  under a random restriction that leaves each variable alive independently with probability  $p$  [SICOMP, 1998]. Using this result, he gave an  $\tilde{\Omega}(n^3)$  formula size lower bound for the Andreev function, which, up to lower order improvements, remains the state-of-the-art lower bound for any explicit function.

In this work, we extend the shrinkage result of Håstad to hold under a far wider family of random restrictions and their generalization – random projections. Based on our shrinkage results, we obtain an  $\tilde{\Omega}(n^3)$  formula size lower bound for an explicit function computed in  $\mathbf{AC}^0$ . This improves upon the best known formula size lower bounds for  $\mathbf{AC}^0$ , that were only quadratic prior to our work. In addition, we prove that the KRW conjecture [Karchmer et al., Computational Complexity 5(3/4), 1995] holds for inner functions for which the unweighted quantum adversary bound is tight. In particular, this holds for inner functions with a tight Khrapchenko bound.

Our random projections are tailor-made to the function's structure so that the function maintains structure even under projection – using such projections is necessary, as standard random restrictions simplify  $\mathbf{AC}^0$  circuits. In contrast, we show that any De Morgan formula shrinks by a quadratic factor under our random projections, allowing us to prove the cubic lower bound.

Our proof techniques build on the proof of Håstad for the simpler case of balanced formulas. This allows for a significantly simpler proof at the cost of slightly worse parameters. As such, when specialized to the case of  $p$ -random restrictions, our proof can be used as an exposition of Håstad's result.

**2012 ACM Subject Classification** Theory of computation → Circuit complexity

**Keywords and phrases** De Morgan formulas, KRW Conjecture, shrinkage, random restrictions, random projections, bounded depth circuits, constant depth circuits, formula complexity

**Digital Object Identifier** 10.4230/LIPIcs.ITCS.2021.89

**Category** Extended Abstract

**Related Version** A full version of the paper is available at <https://yuvalfilmus.cs.technion.ac.il/Papers/shrinkage.pdf>.

**Funding** *Yuval Filmus*: Taub Fellow – supported by the Taub Foundations. The research was funded by ISF grant 1337/16.

*Or Meir*: Partially supported by the Israel Science Foundation (grant No. 1445/16).



© Yuval Filmus, Or Meir, and Avishay Tal;

licensed under Creative Commons License CC-BY

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).

Editor: James R. Lee; Article No. 89; pp. 89:1–89:7

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

**Acknowledgements** A.T. would like to thank Igor Carboni Oliveira for bringing the question of proving formula size lower bounds for  $\text{AC}^0$  to his attention. We are also grateful to Robin Kothari for posing this open question on “Theoretical Computer Science Stack Exchange” [23], and to Kaveh Ghasemloo and Stasys Jukna for their feedback on this question. We would like to thank Anna Gál for very helpful discussions.

## 1 Introduction

### 1.1 Background

Is there an efficient computational task that cannot be perfectly parallelized? Equivalently, is  $\mathbf{P} \not\subseteq \mathbf{NC}^1$ ? The answer is still unknown. The question can be rephrased as follows: is there a function in  $\mathbf{P}$  that does not have a (De Morgan) formula of polynomial size?

The history of formula lower bounds for functions in  $\mathbf{P}$  goes back to the 1960s, with the seminal result of Subbotovskaya [30] that introduced the technique of random restrictions. Subbotovskaya showed that the Parity function on  $n$  variables requires formulas of size at least  $\Omega(n^{1.5})$ . Khrapchenko [22], using a different proof technique, showed that in fact the Parity function on  $n$  variables requires formulas of size  $\Theta(n^2)$ . Later, Andreev [3] came up with a new explicit function (now known as the Andreev function) for which he was able to obtain an  $\Omega(n^{2.5})$  size lower bound. This lower bound was subsequently improved by [18, 25, 14, 31] to  $n^{3-o(1)}$ .

The line of work initiated by Subbotovskaya and Andreev relies on the *shrinkage* of formulas under  $p$ -random restrictions. A  $p$ -random restriction is a randomly chosen partial assignment to the inputs of a function. Set a parameter  $p \in (0, 1)$ . We fix each variable independently with probability  $1 - p$  to a uniformly random bit, and we keep the variable alive with probability  $p$ . Under such a restriction, formulas shrink (in expectation) by a factor more significant than  $p$ . Subbotovskaya showed that De Morgan formulas shrink to at most  $p^{1.5}$  times their original size, whereas subsequent works of [25, 18] improved the bound to  $p^{1.55}$  and  $p^{1.63}$ , respectively. Finally, Håstad [14] showed that the shrinkage exponent of De Morgan formulas is 2, or in other words, that De Morgan formulas shrink by a factor of  $p^{2-o(1)}$  under  $p$ -random restrictions. Tal [31] improved the shrinkage factor to  $O(p^2)$  – obtaining a tight result, as exhibited by the Parity function.

In a nutshell, shrinkage results are useful to proving lower bounds as long as the explicit function being analyzed maintains structure under such restrictions and does not trivialize. For example, the Parity function does not become constant as long as at least one variable remains alive. Thus any formula  $F$  that computes Parity must be of at least quadratic size, or else the formula  $F$  under restriction, keeping each variable alive with probability  $100/n$ , would likely become a constant function, whereas Parity would not. Andreev’s idea is similar, though he manages to construct a function such that under a random restriction keeping only  $\Theta(\log n)$  of the variables, the formula size should be at least  $\tilde{\Omega}(n)$  (in expectation). This ultimately gives the nearly cubic lower bound.

### The KRW Conjecture

Despite much effort, proving  $\mathbf{P} \not\subseteq \mathbf{NC}^1$ , and even just breaking the cubic barrier in formula lower bounds, have remained a challenge for more than two decades. An approach to solve the  $\mathbf{P}$  versus  $\mathbf{NC}^1$  problem was suggested by Karchmer, Raz and Wigderson [20]. They conjectured that when composing two Boolean functions,  $f$  and  $g$ , the formula size of the

resulting function,  $f \diamond g$ , is (roughly) the product of the formula sizes of  $f$  and  $g$ .<sup>1</sup> We will refer to this conjecture as the “KRW conjecture”. Under the KRW conjecture (and even under weaker variants of it), [20] constructed a function in  $\mathbf{P}$  with no polynomial-size formulas. It remains a major open challenge to settle the KRW conjecture.

A few special cases of the KRW conjecture are known to be true. The conjecture holds when either  $f$  or  $g$  is the AND or the OR function. Håstad’s result [14] and its improvement [31] show that the conjecture holds when the inner function  $g$  is the Parity function and the outer function  $f$  is any function. This gives an alternative explanation to the  $n^{3-o(1)}$  lower bound for the Andreev function. Indeed, the Andreev function is at least as hard as the composition of a maximally-hard function  $f$  on  $\log n$  bits and  $g = \text{Parity}_{n/\log n}$ , where the formula size of  $f$  is  $\tilde{\Omega}(n)$  and the formula size of  $\text{Parity}_{n/\log n}$  is  $\Theta(n^2/\log^2 n)$ . Since the KRW conjecture holds for this special case, the formula size of the Andreev function is at least  $\tilde{\Omega}(n^3)$ . In other words, the state-of-the-art formula size lower bounds for explicit functions follow from a special case of the KRW conjecture – the case in which  $g$  is the Parity function. Moreover, this special case follows from the shrinkage of De Morgan formulas under  $p$ -random restrictions.

### Bottom-Up versus Top-Down Techniques

Whereas random restrictions are a “bottom-up” proof technique [15], a different line of work suggested a “top-down” approach using the language of communication complexity. The connection between formula size and communication complexity was introduced in the seminal work of Karchmer and Wigderson [21]. They defined for any Boolean function  $f$  a two-party communication problem  $KW_f$ : Alice gets an input  $x$  such that  $f(x) = 1$ , and Bob gets an input  $y$  such that  $f(y) = 0$ . Their goal is to identify a coordinate  $i$  on which  $x_i \neq y_i$ , while minimizing their communication. It turns out that there is a one-to-one correspondence between any protocol tree solving  $KW_f$  and any formula computing the function  $f$ . Since protocols naturally traverse the tree from root to leaf, proving lower bounds on their size or depth is done usually in a top-down fashion. This framework has proven to be very useful in proving formula lower bounds in the monotone setting (see, e.g., [21, 10, 28, 20, 27, 11, 26]). Moreover, a recent work by Dinur and Meir [6] was able to reprove Håstad’s cubic lower bound using the framework of Karchmer and Wigderson. As Dinur and Meir’s proof showed that top-down techniques can replicate Håstad’s cubic lower bound, a natural question (which motivated this project) arose:

*Are top-down techniques superior to bottom-up techniques?*

Towards that, we focused on a candidate problem: prove a cubic lower bound for an explicit function in  $\mathbf{AC}^0$ .<sup>2</sup> Based on the work of Dinur and Meir [6], we suspected that such a lower bound could be achieved using top-down techniques. We were also *certain* that the problem cannot be solved using the random restriction technique. Indeed, in order to prove a lower bound on a function  $f$  using random restrictions, one should argue that  $f$  remains hard under a random restriction, however, it is well-known that functions in  $\mathbf{AC}^0$  trivialize under  $p$ -random restrictions [7, 1, 32, 12]. Based on this intuition, surely random restrictions cannot show that a function in  $\mathbf{AC}^0$  requires cubic size. Our intuition turned out to be false.

<sup>1</sup> More precisely, the original KRW conjecture [20] concerns depth complexity rather than formula complexity. The variant of the conjecture for formula complexity, which is discussed above, was posed in [9].

<sup>2</sup> Recall that  $\mathbf{AC}^0$  is the class of functions computed by constant depth polynomial size circuits composed of AND and OR gates of unbounded fan-in, with variables or their negation at the leaves.

## 1.2 Our results

In this work, we construct an explicit function in  $\mathbf{AC}^0$  which requires De Morgan formulas of size  $n^{3-o(1)}$ . Surprisingly, our proof is conducted via the bottom-up technique of random projections, which is a generalization of random restrictions (more details below).

► **Theorem 1.** *There exists a family of Boolean functions  $h_n : \{0, 1\}^n \rightarrow \{0, 1\}$  for  $n \in \mathbb{N}$  such that*

1.  $h_n$  can be computed by uniform depth-4 unbounded fan-in formulas of size  $O(n^3)$ .
2. The formula size of  $h_n$  is at least  $n^{3-o(1)}$ .

Prior to our work, the best formula size lower bounds on an explicit function in  $\mathbf{AC}^0$  were only quadratic [24, 5, 19, 4].

Our hard function is a variant of the Andreev function. More specifically, recall that the Andreev function is based on the composition  $f \diamond g$ , where  $f$  is a maximally-hard function and  $g$  is the Parity function. Since Parity is not in  $\mathbf{AC}^0$ , we cannot take  $g$  to be the Parity function in our construction. Instead, our hard function is obtained by replacing the Parity function with the Surjectivity function of [4].

As in the case of the Andreev function, we establish the hardness of our function by proving an appropriate special case of the KRW conjecture. To this end, we introduce a generalization of the complexity measure of Khrapchenko [22], called the *min-entropy Khrapchenko bound*. We prove the KRW conjecture for the special case in which the outer function  $f$  is any function, and  $g$  is a function whose formula complexity is bounded tightly by the min-entropy Khrapchenko bound. We then obtain Theorem 1 by applying this version of the KRW conjecture to the case where  $g$  is the Surjectivity function. We note that our KRW result also implies the known lower bounds in the cases where  $g$  is the Parity function [14] and the Majority function [8].

Our KRW result in fact applies more generally, to functions  $g$  whose formula complexity is bounded tightly by the “soft-adversary method”, denoted  $\text{Adv}_s(g)$ , which is a generalization of Ambainis’ unweighted adversary method [2].

Our proof of the special case of the KRW conjecture follows the methodology of Håstad [13], who proved the special case in which  $g$  is Parity on  $m$  variables. Håstad proved that De Morgan formulas shrink by a factor of (roughly)  $p^2$  under  $p$ -random restrictions. Choosing  $p = 1/m$  shrinks a formula for  $f \diamond g$  by a factor of roughly  $m^2$ , which coincides with the formula complexity of  $g$ . On the other hand, on average each copy of  $g$  simplifies to a single input variable, and so  $f \diamond g$  simplifies to  $f$ . This shows that  $L(f \diamond g) \gtrsim L(f) \cdot L(g)$ .

Our main technical contribution is a new shrinkage theorem that works in a far wider range of scenarios than just  $p$ -random restrictions. Given a function  $g$  with soft-adversary bound  $\text{Adv}_s(g)$ , we construct a random projection<sup>3</sup> which, on the one hand, shrinks De Morgan formulas by a factor of  $\text{Adv}_s(g)$ , and on the other hand, simplifies  $f \diamond g$  to  $f$ . We thus show that  $L(f \diamond g) \gtrsim L(f) \cdot \text{Adv}_s(g)$ , and in particular, if  $\text{Adv}_s(g) \approx L(g)$ , then  $L(f \diamond g) \gtrsim L(f) \cdot L(g)$ , just as in Håstad’s proof. Using these random projections, that are tailored specifically to the structure of the function  $f \diamond g$  so that  $f \diamond g$  simplifies to  $f$  under projection, enables us to overcome the aforementioned difficulty. In contrast,  $p$ -random restrictions that do not respect the structure of  $f \diamond g$  would likely result in a restricted function that is much simpler than  $f$  and in fact would be a constant function with high probability.

<sup>3</sup> A projection is a mapping from the set of the variables  $\{x_1, \dots, x_n\}$  to the set  $\{y_1, \dots, y_m, \overline{y_1}, \dots, \overline{y_m}, 0, 1\}$ , where  $y_1, \dots, y_m$  are formal variables.

Our shrinkage theorem applies more generally to two types of random projections, which we call *fixing projections* and *hiding projections*. Fixing projections are random projections in which fixing the value of a variable results in a projection which is much more probable. Hiding projections are random projections in which fixing the value of a variable hides which coordinates it appeared on. We note that our shrinkage theorem for fixing projections captures Håstad's result for  $p$ -random restrictions as a special case.

The proof of our shrinkage theorem is based on Håstad's proof [14], but also simplifies it. In particular, we take the simpler argument that Håstad uses for the special case of completely balanced trees, and adapt it to the general case. As such, our proof avoids a complicated case analysis, at the cost of slightly worse bounds. Using our bounds, it is nevertheless easy to obtain the  $n^{3-o(1)}$  lower bound for the Andreev function. Therefore, one can see the specialization of our shrinkage result to  $p$ -random restrictions as an exposition of Håstad's cubic lower bound.

### An example: our techniques when specialized to $f \diamond \text{Majority}_m$

To illustrate our choice of random projections, we present its instantiation to the special case of  $f \diamond g$  where  $f : \{0, 1\}^k \rightarrow \{0, 1\}$  is non-constant and  $g = \text{Majority}_m$  for some odd integer  $m$ . In this case, the input variables to  $f \diamond g$  are composed of  $k$  disjoint blocks,  $B_1, \dots, B_k$ , each containing  $m$  variables. We use the random projection that for each block  $B_i = \{x_{m(i-1)+1}, \dots, x_{mi}\}$ , picks one variable in the block  $B_i$  uniformly at random, projects this variable to the new variable  $y_i$ , and fixes the rest of the variables in the block in a balanced way so that the number of zeros and ones in the block is equal (i.e., we have exactly  $(m-1)/2$  zeros and  $(m-1)/2$  ones). It is not hard to see that under this choice,  $f \diamond g$  simplifies to  $f$ . On the other hand, we show that this choice of random projections shrinks the formula complexity by a factor of  $\approx 1/m^2$ . Combining the two together, we get that  $L(f \diamond \text{Majority}_m) \gtrsim L(f) \cdot m^2$ . Note that in this distribution of random projections, the different coordinates are not independent of one another, and this feature allows us to maintain structure.

## 1.3 Related work

Our technique of using tailor-made random projections was inspired by the celebrated result of Rossman, Servedio, and Tan [29, 16] that proved an average-case depth hierarchy. In fact, the idea to use tailor-made random restrictions goes back to Håstad's thesis [17, Chapter 6.2]. Similar to our case, in [17, 29, 16],  $p$ -random restrictions are too crude to separate depth  $d$  from depth  $d+1$  circuits. Given a circuit  $C$  of depth  $d+1$ , the main challenge is to construct a distribution of random restrictions or projections (tailored to the circuit  $C$ ) that on the one hand maintains structure for  $C$ , but on the other hand simplify any depth  $d$  circuit  $C'$ .

### Full Version

Due to space constraints, we have only included in this extended abstract the introduction of our paper. We defer the reader to the full version of the paper for more details and complete proofs.

## References

- 1 Miklós Ajtai.  $\Sigma_1^1$ -formulae on finite structures. *Annals of Pure and Applied Logic*, 24(1):1–48, 1983.
- 2 Andris Ambainis. Quantum lower bounds by quantum arguments. *J. Comput. System Sci.*, 64(4):750–767, 2002. Special issue on STOC 2000 (Portland, OR). doi:10.1006/jcss.2002.1826.
- 3 Alexander E. Andreev. On a method for obtaining more than quadratic effective lower bounds for the complexity of  $\pi$ -schemes. *Moscow University Mathematics Bulletin*, 42(1):24–29, 1987.
- 4 Paul Beame and Widad Machmouchi. The quantum query complexity of AC0. *Quantum Info. Comput.*, 12(7–8):670–676, July 2012.
- 5 Andrew M. Childs, Shelby Kimmel, and Robin Kothari. The quantum query complexity of read-many formulas. In Leah Epstein and Paolo Ferragina, editors, *Algorithms – ESA 2012*, pages 337–348, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.
- 6 Irit Dinur and Or Meir. Toward the KRW composition conjecture: Cubic formula lower bounds via communication complexity. *Computational Complexity*, 27(3):375–462, 2018.
- 7 Merrick L. Furst, James B. Saxe, and Michael Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical Systems Theory*, 17(1):13–27, 1984.
- 8 Anna Gál, Avishay Tal, and Adrian Trejo Nuñez. Cubic formula size lower bounds based on compositions with majority. In Avrim Blum, editor, *10th Innovations in Theoretical Computer Science Conference, ITCS 2019, January 10–12, 2019, San Diego, California, USA*, volume 124 of *LIPIcs*, pages 35:1–35:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.ITCS.2019.35.
- 9 Dmitry Gavinsky, Or Meir, Omri Weinstein, and Avi Wigderson. Toward better formula lower bounds: The composition of a function and a universal relation. *SIAM J. Comput.*, 46(1):114–131, 2017.
- 10 Mikael Goldmann and Johan Håstad. A simple lower bound for monotone clique using a communication game. *Inf. Process. Lett.*, 41(4):221–226, 1992.
- 11 Mika Göös and Toniann Pitassi. Communication lower bounds via critical block sensitivity. *SIAM J. Comput.*, 47(5):1778–1806, 2018.
- 12 Johan Håstad. Almost optimal lower bounds for small depth circuits. In *STOC*, pages 6–20, 1986.
- 13 Johan Håstad. The shrinkage exponent is 2. In *Proceedings of the 1993 IEEE 34th Annual Foundations of Computer Science, SFCS '93*, pages 114–123, USA, 1993. IEEE Computer Society. doi:10.1109/SFCS.1993.366876.
- 14 Johan Håstad. The shrinkage exponent of de Morgan formulas is 2. *SIAM J. Comput.*, 27(1):48–64, 1998.
- 15 Johan Håstad, Stasys Jukna, and Pavel Pudlák. Top-down lower bounds for depth-three circuits. *Computational Complexity*, 5(2):99–112, 1995.
- 16 Johan Håstad, Benjamin Rossman, Rocco A. Servedio, and Li-Yang Tan. An average-case depth hierarchy theorem for boolean circuits. *J. ACM*, 64(5):35:1–35:27, 2017. doi:10.1145/3095799.
- 17 Johan Torkel Håstad. *Computational limitations for small-depth circuits*. MIT press, 1987.
- 18 Russell Impagliazzo and Noam Nisan. The effect of random restrictions on formula size. *Random Struct. Algorithms*, 4(2):121–134, 1993.
- 19 Stasys Jukna. *Boolean Function Complexity - Advances and Frontiers*, volume 27 of *Algorithms and combinatorics*. Springer, 2012.
- 20 Mauricio Karchmer, Ran Raz, and Avi Wigderson. Super-logarithmic depth lower bounds via the direct sum in communication complexity. *Computational Complexity*, 5(3/4):191–204, 1995.
- 21 Mauricio Karchmer and Avi Wigderson. Monotone circuits for connectivity require super-logarithmic depth. *SIAM J. Discrete Math.*, 3(2):255–265, 1990.
- 22 V. M. Khrapchenko. A method of obtaining lower bounds for the complexity of  $\pi$ -schemes. *Mathematical Notes Academy of Sciences USSR*, 10:474–479, 1972.



- 23 Robin Kothari. Formula size lower bounds for AC0 functions, 2011. Question on Theoretical Computer Science Stack Exchange. URL: <https://cstheory.stackexchange.com/questions/7156/formula-size-lower-bounds-for-ac0-functions>.
- 24 E. I. Neciporuk. On a Boolean function. *Soviet Mathematics Doklady*, 7(4):999–1000, 1966.
- 25 Mike Paterson and Uri Zwick. Shrinkage of de Morgan formulae under restriction. *Random Struct. Algorithms*, 4(2):135–150, 1993.
- 26 Toniann Pitassi and Robert Robere. Strongly exponential lower bounds for monotone computation. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 1246–1255, 2017.
- 27 Ran Raz and Pierre McKenzie. Separation of the monotone NC hierarchy. *Combinatorica*, 19(3):403–435, 1999.
- 28 Ran Raz and Avi Wigderson. Monotone circuits for matching require linear depth. *J. ACM*, 39(3):736–744, 1992.
- 29 Benjamin Rossman, Rocco A. Servedio, and Li-Yang Tan. An average-case depth hierarchy theorem for Boolean circuits. In *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 1030–1048, 2015.
- 30 Bella Abramovna Subbotovskaya. Realizations of linear functions by formulas using +, ·, -. *Soviet Mathematics Doklady*, 2:110–112, 1961.
- 31 Avishay Tal. Shrinkage of de Morgan formulae by spectral techniques. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 551–560, 2014.
- 32 Andrew Chi-Chih Yao. Separating the polynomial-time hierarchy by oracles (preliminary version). In *FOCS*, pages 1–10. IEEE Computer Society, 1985.

